Mensile - 25 Febbraio 1986 - Anno V - N. 28 - Sped. Abb. Post. Gr. III/70 - CR - Distr.: MePe

C 64 e Plus 4 a confronto

Come costruire un rack o un pulsante di reset

La moltiplicazione delle tracce e dei settori

Lente d'ingrandimento per C 64

Systems

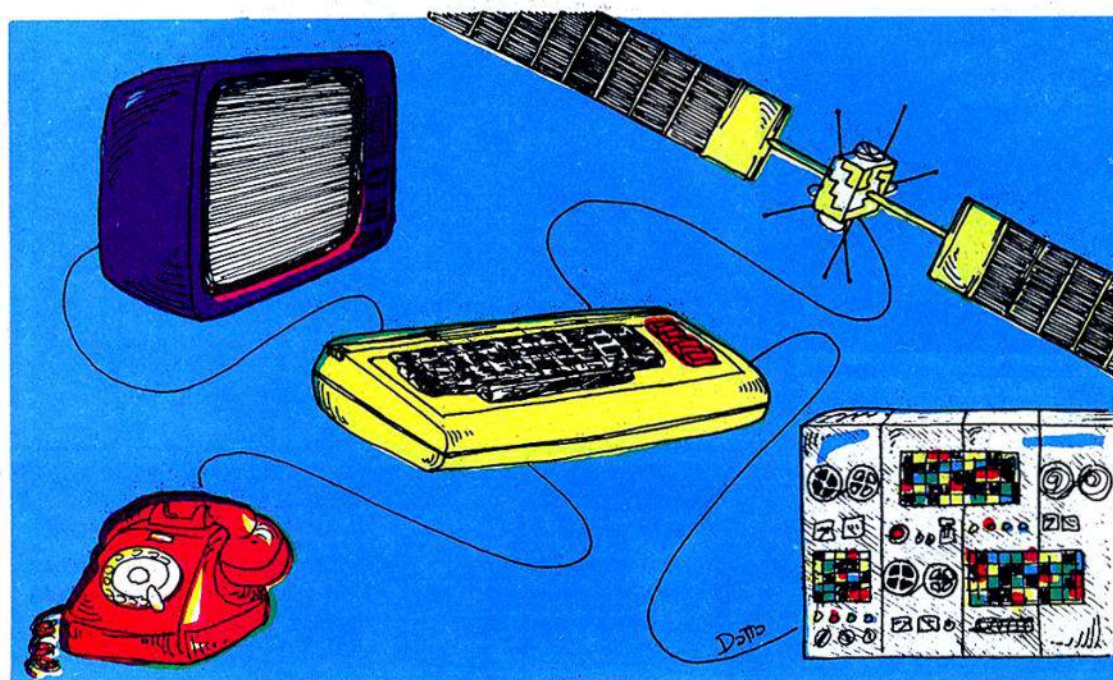
C 16, ed è subito grafica

Sommario

RUBRICHE

Direttore: Alessandro de Simone
Redazione/collaboratori: Carlo e Lorenzo Barazzetta, Giovanni Bellù, Simone Bettola, Andrea e Alberto Boriani, Diego e Federico Canetta, Giancarlo Castanga, Eugenio Coppari Pasquale D'Andreti, Maurizio Dell'Abate, Marco De Martino, Luca Galluzzi, Michele Maggi, Giancarlo Mariani, Flavio Molinari, Claudio Mueller, Enrico Scelsa, D. Matturro, M.L. Nitti, Massimo Pollutri, Carla Rampi, Fabio Sorgato, Giovanni Verrelli, Antonio Visconti.
Segreteria di redazione: Maura Ceccaroli, Piera Perin
Ufficio Grafico: Mary Benvenuto, Arturo Ciaglia, Paolo Vertuccio
Direzione, redazione, pubblicità: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Leandro Nencioni (direttore vendite), Giorgio Ruffoni, Roberto Sghirinzetti (settore informatica), Claudio Tidone - V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
● Emilia Romagna: Spazio E - P.zza Roosevel, 4 - 40123 Bologna - Tel. 051/236979
● Toscana, Marche, Umbria: Mercurio Srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
● Lazio, Campania: Spazio Nuovo - via P. Foscari 70 - 00139 Roma - Tel. 06/8109679
Segretaria: Lilliana Degiorgi - **Abbonamenti:** Marina Vantini
Tariffe: prezzo per copia L. 3.500. Abbonamento annuo (11 fascicoli) L. 35.000. Estero: il doppio. Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 70.000.
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizioni: Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
Stampa: La Litografica S.r.l. - Busto Arsizio (VA)
Registrazione: Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib:** MePe, via G. Carcano 32 - Milano

Oltre il computer, verso la fantasia

Il possesso del tuo home computer non è che il primo passo verso il mondo affascinante dell'informatica. La tastiera, che non smette di stupirti, va considerata come la punta di un immenso iceberg misterioso. Ti aiuteremo a scendere nelle profondità del tuo calcolatore, dentro i chip, a contatto con la nuova Fantasilandia tecnologica....

Tempo fa, recatomi a Ginevra su invito di una multinazionale di cui non posso dire il nome ma solo le iniziali (IBM), mi resi conto che il vero futuro dell'informatica non risiede affatto nell'oggetto "computer" ma nel servizio che questo può offrire e, in particolare, nelle telecomunicazioni.

Nel mondo di domani (e non intendo il 2100, ma mi riferisco al prossimo quinquennio), sarà privilegiato chi disporrà di servizi affidabili di invio di dati.

Pensate agli uffici e, in particolare, a quelli che hanno più sedi distanti tra loro magari centinaia di chilometri. Quando si verifica la necessità di disporre in tempi brevi di informazioni varie (documenti, lettere, ordini di merce, controlli vari, eccetera) è possibile, oggi come oggi, ricorrere a sistemi a dir poco folcloristici:

- il servizio postale di Stato, la cui rapidità ed efficienza superano, a volte, persino quelle registrate durante le invasioni barbariche, non sembrano incontrare il favore di molti utenti;

- i corrieri privati, in grado di garantire la consegna di pacchi e lettere in tempi brevi, chiedono giustamente (!) un alto prezzo ma, nonostante ciò, possono presentare limiti vari;

- le ferrovie di Stato, prodigio di puntualità, permettono avventurose spedizioni e mantengono il destinatario del pacco in costante trepidazione (verrà? non verrà?...);

- i telefoni (sempre di Stato) consentono l'invio di segnali acustici e addirittura, adeguandosi agli standard introdotti dopo la scoperta dell'elettricità, sembra che alcune linee siano talmente "pulite" ed esenti da disturbi che è possibile (udite, udite) inviare (e, assicurano, ricevere) segnali digitali generati da computer o similari.

Vogliamo fermarci un attimo e considerare (seriamente) che cosa ci offre il futuro delle telecomunicazioni?

- Videotelefono. Benchè possa sembrare un vezzo, l'invio contemporaneo di segnali acustici e video (magari a colori) consentirà una maggior efficienza del posto di lavoro.

Senza spostarvi dalla scrivania, infatti, potrete inviare disegni, mostrare documenti speciali, controllare (anche se superficialmente) la firma di una lettera eccetera. Per telefono, oggi, è possibile esprimersi soltanto a voce e, molto spesso, questo sistema non è sufficiente per la descrizione di un documento prevalentemente grafico.

La possibilità di digitalizzare un'immagine e la conseguente manipolazione (ingrandimenti, selezione dei colori, archiviazione) non renderà più necessario l'invio "fisico" di un documento.

Ciò si traduce in un considerevole risparmio di tempo, di mezzi di trasporto, di assenze "forzate" dal posto di lavoro che, oggi, si rendono indispensabili per recarsi di persona a consegnare (ed illustrare) i documenti.

- Sempre grazie allo sviluppo delle telecomunicazioni sarà una realtà quotidiana il servizio di tele-medicina che permetterà al centro ospedaliero (e non è fantascienza) di esaminarvi via telefono (o strumento similare). Sarà sufficiente collegare una serie di sensori al vostro corpo e stabilire se è il caso di inviare uno specialista al vostro indirizzo per una visita più accurata. E' ovvio che il futuro (magari un po' più remoto) consentirà di eliminare le degenze ospedaliere dei pazienti che potranno essere controllati a casa propria via cavo.

Ciò si potrà tradurre, come è intuibile, in una diminuzione dei costi sociali e, soprattutto, in minori disagi per il paziente e i suoi familiari.

- Le superaffollate scuole e università, oltre alle società private, potranno usufruire delle videoconferenze. Queste, permettendo lo svolgimento di un dibattito tra due gruppi di partecipanti (anche a distanze intercontinentali), consentiranno la divulgazione di notizie, l'aggiornamento professionale, la rapidità di decisioni in questioni aziendali. E tutto questo senza, beninteso, costringere i due gruppi a incontrarsi da vicino col conseguente risparmio delle spese per il viaggio, per la sistemazione in albergo e per la sospensione prolungata dell'attività lavorativa.

Che rapporto ha tutto questo con Commodore Computer Club? Che cosa hanno a che fare le telecomunicazioni con i videogiochi, i programmi di simulazione, la gestione del disco?

Ma è evidente!

L'utente del piccolo computer di oggi sarà il cittadino di domani che, in un mondo tecnologicamente più evoluto, dovrà destreggiarsi tra protocolli d'interfaccia, programmi di comunicazioni, problemi di invio dati lungo linee telefoniche e (perchè no?) via satellite. E al centro di tutto questo c'è il cosiddetto hardware, i circuiti integrati, i transistors, i connettori per collegare più computer tra loro.

Per questo, a partire da questo numero, cercheremo di stimolare il lettore ad andare oltre il piccolo pianeta del suo modesto computer, accompagnandolo in un viaggio meraviglioso.

Dapprima costruiremo semplici oggetti da affiancare al calcolatore, da paragonare a piccoli voli di prova, come quelli dei fratelli Whrite. Poi ci renderemo conto, insieme, che lavorare col saldatore non è così difficile (nè pericoloso). In seguito, dopo lo Space Shuttle dei primi minuscoli e divertenti circuiti elettronici da collegare al vostro Commodore, ci leveremo in alto, verso altri pianeti, verso il sole vero dell'informatica.

E, stavolta, non ci saranno ali che rischiano di bruciarsi...

Alessandro de Simone

Neutralizzazione del tasto Restore

☐ **Come aggiungere la possibilità di disattivare il tasto Restore nelle routine grafiche su nastro di Commodre Club N.4?** (numerosi lettori)

• Per disattivare il tasto Restore (vale sempre l'ipotesi che disponiate solo della versione nastro delle routine grafiche), potete aggiungere al listato di GRSAVE & C. le seguenti linee:

```
10000 FOR I=49874 TO 49887
:READA :POKE I,A :NEXT

10100
DATA133,1,169,223,141,250,255,
169,194,141,251,255,96,64
```

che sono gli stessi DATA proposti nel punto 4 dell'articolo del numero 18 di C.C.C.

```
10200 POKE 50538,76: POKE
50539,210: POKE 50540,194
```

E' tutto.

Hard copy e Simon's basic

☐ **La routine di copia su carta (COPY) della pagina grafica non funziona col Simon's BASIC e la stampante MPS 802. Come mai?** (Angelo Pagano - Bari)

• Il programma Simon's Basic per il Commodore 64 fu realizzato quando in commercio figurava la sola stampante MPS 801 che ha un modo totalmente diverso di gestire la grafica in alta risoluzione. E' ovvio che il programma funziona correttamente con la 801.

Su un numero precedente di C.C.C. (ed esattamente quello di febbraio - marzo) abbiamo pubblicato una routine di hard copy che funziona anche col Simon's BASIC.

Incompatibilità di dischi

☐ **I dischetti doppia densità sono incompatibili col drive 1541? In caso negativo, perchè non riesco ad usarli col mio drive?** (Maurizio Salzano - Capri)

• Il drive 1541 si "accontenta" di dischetti a singola faccia e singola densità, dato che, per come è costruito, non può garantire una compattazione di dati superiore ad un certo livello.

I dischetti a doppia densità sono di qualità migliore (i più esperti perdonino questo linguaggio terra terra) e, di conseguenza, sprecati con un drive 1541.

Se non riesci a leggere un dischetto a doppia densità è molto probabile che questo non sia stato scritto da un drive 1541, oppure vi sono file o programmi memorizzati ricorrendo a particolari programmi di utilità che aumentano artificiosamente il numero di tracce e settori. Programmi scritti ricorrendo a tali utility possono esser letti solo ricorrendo alle stesse utility.

Espansioni per C-16

☐ **E' possibile espandere la memoria del Commodore 16?** (Patrik Ronco - Milano)

• La quantità di memoria gestibile da un microprocessore della famiglia 65XX (come quello del C-16) è data dalla sedicesima potenza di due: 65536. Poichè la RAM disponibile è di "soli" 16K sarebbe possibile, in teoria, portarla a 64 K RAM.

Purtroppo è necessario, oltre ad aggiungere la scheda elettronica contenente la RAM aggiuntiva,,apportare alcune modifiche al circuito elettrico del computer. Pertanto è necessario consegnarlo al centro specializzato in contatto col rivenditore di espansioni.

Stampa a 132 colonne

☐ **Avete pubblicato un programma che consenta la stampa su 132 colonne anche per le stampanti MPS/802, 803?** (Francesco Niccolai - Parma)

• Sul N.15 (novembre '84) di C.C.C. è stata pubblicata una simpatica (e brevissima) routine in grado di realizzare la funzione richiesta con la stampante MPS/801. L'argomento, che suscitò molti consensi a discapito della relativa lentezza di stampa, non è stato più affrontato ed anzi cogliamo l'occasione per invitare i lettori capaci di intervenire sul listato in questione (scritto in parte in L.M.) in modo da apportare le dovute modifiche per la MPS/803. Un listato idoneo a funzionare sulla MPS/802, se dovesse pervenire in Redazione, sarà pubblicato egualmente, nonostante tale stampante sia ormai uscita di produzione.

String too long error

☐ **In un mio programma che crea (ed in seguito legge) file di dati, mi capita spesso che, in fase di lettura, compaia il messaggio STRING TOO LONG ERROR. A che cosa può esser dovuto?** (Maurizio Epifano - Pescara)

• Come certamente saprai, è possibile creare stringhe piuttosto lunghe (fino a 255 caratteri) ricorrendo a concatenazioni di caratteri, altre stringhe, eccetera.

Mentre, però, è lecita un'operazione di "riversamento" su supporto magnetico (nastro o disco che sia) di stringhe così lunghe, non è possibile utilizzare un'istruzione del tipo INPUT per leggere il dato.

L'istruzione INPUT, infatti, utilizzata per acquisire dati da tastiera o da supporto magnetico (e perfino da interfaccia) accetta al massimo una sequenza di ottanta caratteri.

Mentre, però, è impossibile digitare più di 80 caratteri da tastiera (una riga virtuale è formata da due righe di schermo, cioè 80 caratteri), cercando di "forzare" stringhe più lunghe da periferica, il riempimento del buffer dedicato alla ricezione dei dati impone l'emissione del messaggio ed il blocco del programma.

Per leggere stringhe molto lunghe è pertanto necessario ricorrere all'istruzione GET # che, capace di leggere un solo carattere alla volta, può immagazzinare (e scaricare su video, stampante, matrice, eccetera) il contenuto di un file che occupi anche un intero dischetto (170000 caratteri).

Per evitare di ricorrere al GET non c'è che un sistema. Evitare, in fase di registrazione, che i record superino gli ottanta caratteri ciascuno.

Routine grafiche della cassetta N. 4 di Commodore Club

☐ **Alcuni lettori, che non hanno copiato il listato delle routine grafiche pubblicato sul n. 14, ma che possiedono la versione compattata su cassetta dello stesso, hanno chiesto di aggiungere i comandi GRLOAD, GRSAVE, ecc., (listato su C.C.C. n. 17) a tale versione.**

• L'operazione è semplicissima:
1/ Caricate le routine grafiche di Commodore Club N.4 e date il RUN.
2/ Caricate il listato del programma "comandi per il caricamento/salvataggio della pagina grafica" pubblicato sul n. 17 di C.C.C. (le linee di REM sono superflue) e date il RUN.
3/ Eseguite POKE 51051,13.

La locazione 51051 indica il numero di nuovi comandi attivati che passa da 9 a 13.

Analizzatore sonoro

☐ **Esiste un kit hard/software analizzatore di spettro per il Commodore 64?** (Fabio De Stefani - Roma)

● Un analizzatore di spettro è un particolare apparecchio che, per mezzo di un microfono, è in grado di tracciare su uno schermo, mediante una serie di barre, l'intensità sonora presente nell'ambiente in modo proporzionale alla frequenza di emissione. Se, ad esempio, si suona uno alla volta ciascun tasto di un pianoforte (posto vicino al microfono), si noterà, sullo schermo, che le barre "crescono" da sinistra a destra man mano che la nota suonata si eleva di tono.

Mediante un computer è possibile realizzare un apparecchio del genere che, grazie a un particolare circuito elettronico che trasformi le onde sonore in impulsi digitali, visualizzi sullo schermo barre colorate.

Commodore Computer Club sta prendendo in seria considerazione la possibilità di pubblicare, tra gli altri, anche lo schema di questo apparecchio, a patto che la richiesta dei lettori nel campo hardware sia sufficientemente elevata.

Avanzamenti strani

● La matrice del carattere di una stampante può essere di vari formati. Nella MPS-802, ad esempio, è del formato 8x8 e, quando si invia il comando di "a capo", viene calcolato lo spazio necessario di interlinea in modo da rendere leggibili due righi di scrittura successivi.

Programmando opportunamente il numero di linee per pollice è anche possibile sovrapporre parzialmente due righe successive ma, francamente, non riusciamo a capire l'utilità di una simile impostazione. Leggendo attentamente il libretto di istruzioni della stampante dovresti essere in grado di ridurre al minimo la distanza fra due righe. Nel caso in cui l'interlinea minima non ti soddisfi, non c'è altro sistema per risolvere il problema, a meno che non voglia disassemblare la ROM della stampante, interpretarla, sostituirla con una EPROM con un dispendio di tempo e di energia decisamente oneroso.

Istruzioni incomplete

☐ **In alcuni programmi compare, ad esempio, un'istruzione del tipo:**
180 IF A THEN 5800.

Non vi sembra che manchino alcune istruzioni? La riga riportata ad esempio non dovrebbe essere completata nel modo seguente:
180 IF A=56 THEN GOTO 5800 (Numerosi lettori)

● L'apparente incompletezza della riga si giustifica in base alle seguenti considerazioni:
- quando il computer incontra l'istruzione di salto condizionato IF A THEN... ragiona pressappoco così: nel caso in cui la variabile "A" esiste (cioè è diversa da zero), esegui le istruzioni successive alla parola THEN. In altre parole le due espressioni:

IF A THEN...
IF A< >0 THEN...

sono perfettamente identiche ma, nel primo caso, non solo si risparmia spazio sulla riga Basic, ma si aumenta la velocità di esecuzione del programma.
- Dopo la parola THEN è possibile sottintendere il comando GOTO. Pertanto le due espressioni:

THEN GOTO 5800
THEN 5800

sono identiche. Ciò vale, però, solo nel caso del comando GOTO. Il comando GOSUB è necessario scriverlo per intero.

C-64 made in Usa

☐ **E' noto che un computer americano non può funzionare in Italia a causa del diverso standard adottato in America (NTSC) e nel nostro paese (PAL). Smontando il mio C-64, però, ho notato che vi è un ponticello con le indicazioni PAL NTSC. Non sarebbe sufficiente, in un esemplare americano, staccare il filo da NTSC e saldarlo invece su PAL per vederlo funzionare da noi?** (Francesco Messineo - Reggio Calabria)

● Purtroppo il ponticello che hai notato è solo una delle numerose differenze esistenti nei due modelli. Del resto a che serve acquistare il C-64 negli Usa? Se confronti i prezzi praticati in Italia e quelli delle riviste americane puoi notare che le differenze sono piuttosto esigue. Inoltre tieni presente che:
- l'alimentatore Usa è a 125 volt e ne occorre invece uno a 220;
- la garanzia americana non vale in Italia;
- se, al controllo doganale, si accorgono dell'insolita "importazione", rischi una grossa multa o, nel migliore dei casi, il pagamento dell'Iva e di altre tasse, balzelli e vessazioni varie.

Ne vale la pena?

Espansione per Vic 20

☐ **Nel Vic 20, con una espansione da 16 oppure 32K è possibile caricare programmi che richiedono l'espansione da 3K?**

Con l'espansione da 16K è possibile caricare programmi del C-16?

Come posso sapere se, per un programma su nastro, è necessaria una espansione? (Andrea Salvatori)

● Inserendo l'espansione da 3K nel Vic 20, l'area RAM destinata al Basic viene automaticamente posizionata a partire dalla locazione 1024 fino a 7679, anzichè da 4096 a 7679 (configurazione standard senza alcuna espansione). Inserendo un'espansione di 8, 16 oppure 24K l'indirizzo di partenza si modifica (4608), mentre quello finale, ovviamente, cambia a seconda del tipo di espansione, esattamente:
Scheda da 8K: 4608 - 16383
Scheda da 16K: 4608 - 24575
Scheda da 24K: 4608 - 32767

In linea di massima un programma scritto in Basic (che non contenga istruzioni SYS, POKE, PEEK) funziona normalmente anche se è stato scritto con un'espansione da 3K. Putroppo molti giochi, che richiedono l'espansione da 3K, contengono istruzioni che si riferiscono proprio alla zona RAM (1024-4096) che è assente nelle espansioni maggiori di 8, 16 e 24K. In casi come questo, come puoi intuire, tali programmi non possono girare.

Inserendo un'espansione da 16K nel Vic 20, questo non "diventa" un C-16. I due computer, infatti, sono del tutto incompatibili e diversi sotto tutti i punti di vista. Solo alcuni programmi Basic possono girare sui due computer e, per di più, è necessario leggerli mediante dischetto dato che il sistema di registrazione dei programmi è diverso.

Per sapere se è necessaria o meno un'espansione puoi comportarti nel modo seguente:
- se, caricando il programma senza l'espansione inserita compaiono, a un certo punto, caratteri strani, vuol dire che è necessaria;
- se, con un'espansione inserita, compare il messaggio OUT OF MEMORY ERROR, vuol dire che è necessaria un'espansione di maggior dimensione;
- se il programma viene caricato normalmente ma, dato il RUN, blocca il computer, vuol dire che è necessaria l'espansione diversa (da 3K oppure da 16K).

Tieni presente,comunque, che i programmi normalmente in commercio specificano, sulla confezione, l'espansione necessaria per il loro corretto funzionamento.

Tasto Reset inefficace

☐ **Nonostante l'inserimento di un tasto di Reset, alcuni programmi, come il Soccer, non vengono affatto resettati, ma "ripartono" nuovamente. Come mai?** (Alessandro Risso - Genova)

● Il motivo è da ricercarsi nella particolare protezione inserita (di cui parleremo in uno dei prossimi numeri). Per ora ti basti sapere che, al momento del RUN, il programma provvede a copiare la ROM finale nella RAM sottostante e a modificare il vettore di Reset. Mettendo a massa il pin di Reset, pertanto, il microprocessore "salta" a eseguire la routine specificata negli indirizzi RAM (e non più ROM) che, modificati opportunamente, hanno il nuovo compito di far ripartire il programma dall'inizio, senza interromperlo affatto.

Programmi dagli Stati Uniti

☐ **Vorrei sapere se vi è incompatibilità tra il software acquistato in America e gli apparecchi venduti in Italia.**

● L'incompatibilità tra computer NTSC (americani) e PAL (italiani) risiede, oltre che nel diverso sistema di uscita video, solo nella diversa alimentazione elettrica. Per il software, quindi, nessun problema.

Duplicatori o bidonate?

☐ **Un negoziante mi ha detto che il duplicatore di nastri RCP da voi provato su CCC è una "bidonata" che, tra l'altro, rischia di danneggiare il computer. E' vero?** (Enrico Girardi - Mogliano Veneto)

● Il duplicatore da noi provato è un semplice accessorio che in nessun modo può danneggiare il computer a meno che il secondo registratore sia in corto circuito con la tensione di rete (caso del tutto improbabile). Il preciso funzionamento di tali apparecchietti, al contrario, ci ha fatto desistere dall'inserire protezioni nella nostra rivista su cassetta (Software Club), che risulterebbero, appunto, del tutto inutili. L'unico rischio, a nostro parere, è quello che il negoziante ritiene di correre temendo che la diffusione di questi apparecchi faccia diminuire le sue vendite di programmi su nastro...

Evviva il Vic 20

☐ **Perchè trascurate il Vic 20? Sappiate che noi possessori di questo computer siamo in molti e non lo lasceremo (per il C-64) per ingrassare i venditori della Commodore, che sono legati sicuramente a voi.** (Roberto De Filippo - Merone)

● Mi dispiace davvero che valuti in modo così drastico il (presunto) nostro atteggiamento verso il Vic 20. Ti faccio notare, però, che facciamo il possibile per pubblicare programmi "universali", vale a dire tali che possano girare anche sul Vic 20. E poi, vista la tua passione per questo simpatico computer, perchè non ti impegni e cerchi di adattare i programmi pubblicati al tuo calcolatore? La rubrica "Enciclopedia di routine", "1 riga", "Protezioni" e altre "cose" possono esser ospitate sul Vic 20 con minime modifiche.

Ps: io sono alto e magro.

Simulatore di Spectrum

☐ **Sul numero di luglio/agosto della vostra rivista parlate di un programma che trasforma il C-64 in uno Specrum Sinclair. Di che si tratta e come è possibile averlo?** (Numerosissimi lettori).

● Non pensavamo che quella notizia destasse l'interesse di un così elevato numero di lettori. Possiamo solo dire che, agli inizi dell'85, ci capitò di provare quel programma che, almeno per ciò che riguardava il Basic e le operazioni di caricamento e registrazione dei programmi, rendeva realmente identici i due computer tranne che per la velocità di esecuzione dei programmi. Non era necessario alcun apparecchio esterno per farlo funzionare e il risultato, in effetti, era simpatico: si inseriva un programma registrato con uno Spectrum e lo si caricava e faceva girare normalmente!

Altre versioni di questo simulatore di Spectrum sono state realizzate col passare del tempo e non è assolutamente difficile trovarle in giro. Avete provato a contattare i nostri lettori nella rubrica "Piccoli Annunci"? E' fatta apposta...

Si tenga presente, comunque, che la redazione non vende il programma in oggetto, per nessun motivo, dato che non è in possesso dei diritti d'autore.

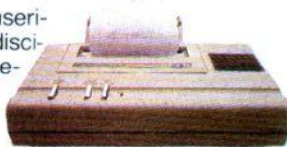
VINCE CHI LEGGE.

Parte l'Operazione Fedeltà Systems. I premi?
Tanti: una moto Cagiva Electra 125, printer-plotter e stampanti a margherita Commodore, abbonamenti alla tua rivista Systems preferita e libri della biblioteca informatica Systems. Partecipare è semplicissimo.

Raccogli i bollini dell'Operazione Fedeltà Systems che troverai

OPERAZIONE FEDELTÀ SYSTEMS.

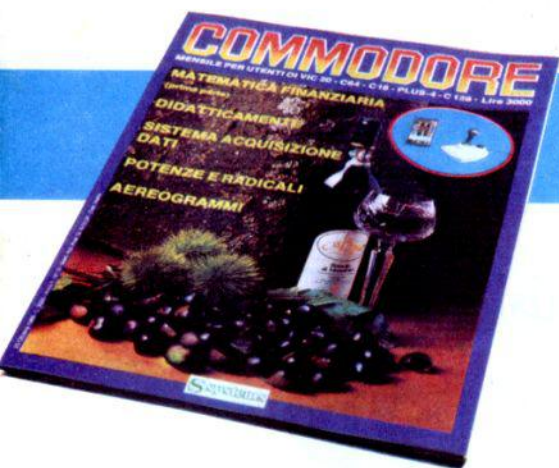
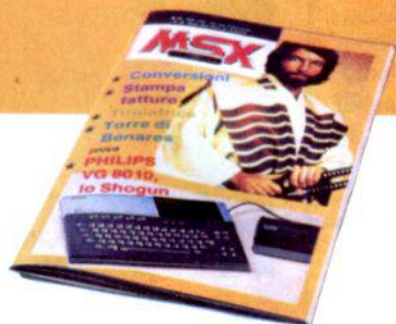
da oggi fino al 31 marzo 1986 - su Commodore Computer Club, Computer, Commodore, Commodore Club, MSX, Personal Computer, Sinclair Computer, 16/48, Special Systems, VR Videoregistrare e sui libri Systems.

incollali sulla cartolina inserita in ogni rivista e spediscila. Ogni mese parteciperanno all'estrazione tutte le cartoline con almeno 5 punti. E ricorda: l'abbonamento a qualsiasi rivista Systems vale subito 5 punti. Inoltre tutte le cartoline parteciperanno al grande concorso finale: in palio una potente moto Cagiva Electra 125.

REGOLAMENTO DEL CONCORSO

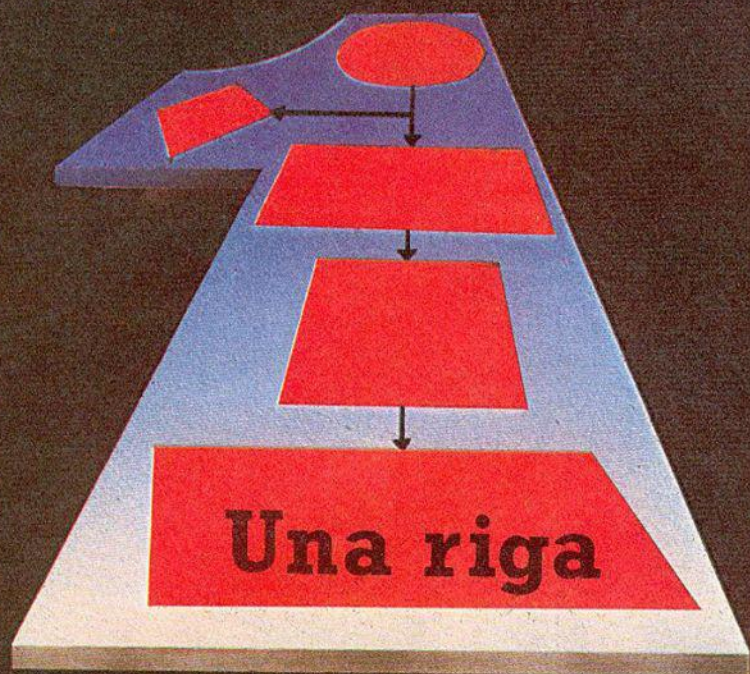
Partecipano all'estrazioni mensili dell'OPERAZIONE FEDELTA' SYSTEMS tutte le cartoline pervenute con almeno 6 punti.
L'abbonamento a una qualsiasi rivista vale 5 punti subito.
I bollini che compaiono su Computer, Commodore Club, 16/48, VR Videoregistrare e i libri Systems valgono 2 punti; quelli di Commodore Computer Club, Commodore e Sinclair Computer 1 punto; quelli delle cassette Special Systems valgono 3 punti.
Il monte premi delle estrazioni mensili è così ripartito:
- dal 1° al 5° estratto: 1 stampante Commodore.
- dal 6° al 10° estratto: 1 printer plotter Commodore.
- dal 11° al 40° estratto: un libro a scelta della biblioteca informatica Systems oppure a scelta un abbonamento ad una rivista Systems.
Tutte le cartoline inviate partecipano all'estrazione finale di una moto Cagiva Electra 125 cc.

EDITORIALE SYSTEMS
"Operazione Fedeltà"
V.le Famagosta, 75
20142 Milano

128 KBYTES

SINCLAIR COM

COMMODORE

MSX

=

DI RIVISTA.

Come al solito, ecco i microprogrammi di una sola riga suddivisi in tre categorie:

- "una riga" universali, cioè per qualsiasi computer Commodore;
- listati specifici per il Commodore 64 (e C-128);
- righe che girano soltanto su C-16 oppure Plus-4.

L'autore di tutti i minilistati è ancora Michele Maggi che ha aggiunto alla sua collezione di computer Commodore (Vic 20, C-64, Plus-4) anche il nuovo Commodore 128. Ha promesso di consegnare microlistati specifici anche per quest'ultimo modello...

Per qualsiasi computer Commodore

#1

Area cerchio. Digitato il raggio, si ottiene in risposta la misura della circonferenza e l'area del cerchio.

```
1 INPUT "RAGGIO";R:
  A=(R↑2)*π:PRINT"A
  REA CERCHIO= "A:P
  RINT"CIRCONF.="2*
  π*R: RUN
```

2

Area triangolo. Un'utility geometrica che permette, data la misura della base e dell'altezza, di conoscere l'area di un qualsiasi triangolo.

```
1 INPUT "BASE,ALTEZ
  ZA";B,H:A=B*H/2:P
  RINT"AREA TRIANGO
  LO= "A: RUN
```

3

Orologio. Sfruttando la variabile riservata TI$, verrà visualizzato sullo schermo, in alto a sinistra, un orologio che dovrà però essere regolato prima di lanciare il programma. Ad esempio:
TI$="123055" significherà ore 12, 30 minuti primi e 55 secondi.

```
1 PRINTCHR$(19)LEFT
  $(TI$,2)CHR$(32)M
  ID$(TI$,3,2)CHR$(
  32)RIGHT$(TI$,2):
  GOTO 1
```

4

Colore cursore. Un facile sistema per vedere come si "intonano" i vari (ben 16) colori del cursore con lo sfondo dello schermo. Attenzione a usare correttamente le abbreviazioni dei comandi e a interpretare correttamente i caratteri "speciali" da sostituire ai colori racchiusi tra parentesi quadre!

```
1 A$="[NERO][BIANCO
  ][ROSSO][AZZUR][V
  IOLA][VERDE][BLEU
  ][GIALLO][ARANC][
  MARR][ROSA][GRIGI
  O1][GRIGIO2][VERD
  E2][CELESTE][GRIG
  IO3]":B$="COL.CUR
  SORE": FOR I=1 TO
   16:PRINTMID$(A$,
  I,1)B$"[RVS]"B$:N
  EXT
```

5

Prova GOSUB. Serve per misurare quanti GOSUB "aperti" possono essere sopportati prima del messaggio OUT OF MEMORY ERROR.

```
1 I=I+1:PRINTCHR$(1
  45)I-1" SUBROUTIN
  E POSSIBILI":GOSU
  B 1
```

Per Commodore 64

6

Colori LM/64. Uno schermo tutto colorato grazie a questa piccola routine in linguaggio macchina (LM).

```
1 DATA 166,0,142,32
  ,208,142,33,208,2
  32,76,2,32:FOR I=
  8192 TO 8203:READ
   A:POKE I,A:NEXT:
  SYS8192
```

7

Input senza punto di domanda. Sfruttando la locazione, 19 è possibile ottenere un input senza il tipico punto di domanda.

```
1 POKE 19,64:INPUT
  "DIGITA CARATTERI
   ";A$:POKE 19,0:P
  RINT:PRINT"HAI SC
  RITTO:"A$
```

8

Colore del cursore. La locazione 646 contiene il valore del colore del cursore: un modo diverso, quindi, per cambiarlo a volontà senza ricorrere ai caratteri "speciali" o ai comandi PRINT CHR$(...).

```
1 FOR I=0 TO 15:POK
  E 646,I:PRINT"COL
  ORE CORRISPONDENT
  E AL NUMERO"I:NEX
  T
```

#9

Disabilita tastiera. Pokando il valore nullo nella locazione 649 (dimensione del buffer di tastiera) è possibile bloccare la tastiera. Per riabilitarla premere Run/stop e Restore.

```
1 POKE 649,0:PRINTC
  HR$(147)"PROVA A
  SCRIVERE QUALCOSA
   SE CI RIESCI.."
```

Per Commodore 16

#10

Colori LM. Una piccolissima routine in LM che trasformerà il video in una fantasia di colori.

```
1 S=8192:FORI=0TO12:
2 READA:POKES+I,A:
3 NEXT:SYSS
4 DATA162,0,142,21,255,
142,25,255,232,120,76,
2,32
```

#11

Scala musicale. Una simpatica scala musicale di un'ottava eseguita dal vostro C-16.

```
1 VOL8:FORI=1TO8:
2 READA:SOUND1,A,15:
3 NEXT:
4 DATA596,643,685,704,
739,770,798,810
```

#12

Disable Keys. Questa una riga disabilita i tasti funzione del C-16 o del C-128 rendendoli inattivi come nel C-64.

```
1 FORI=1TO8:
2 KEYI,CHR$(132+I):
3 NEXT
```

#13

Caratteri LM. Quasi istantaneamente i 256 caratteri del vostro C-16 appariranno sullo schermo grazie a questa una riga in LM.

```
1 S=8192:FORI=0TO15:
2 READA:POKES+I,A:
3 NEXT:SYSS:
4 DATA160,0,162,0,152,
157,0,12,232,200,240,
3,76,4,32,96
```

#14

Pianeta. Sfruttando l'istruzione CIRCLE, comparirà un pianeta con tanto di meridiani e paralleli.

```
1 GRAPHIC1,1:
2 FORY=50TO0STEP-5:
3 CIRCLE1,160,100,Y,50:
4 CIRCLE1,160,100,50,Y:
5 NEXT:GETKEYA$:
6 GRAPHIC0
```

Nota Bene

Alcune righe tra quelle pubblicate sembrano possedere più di 80 caratteri e, come tali, inaccettabili dal computer.

Nei casi in cui ci si accorga che la riga è troppo lunga, è necessario ricorrere alle abbreviazioni dei comandi così come indicato nell'appendice specifica riportata nel manuale del computer in vostro possesso.

Ad esempio invece di scrivere PRINT è possibile abbreviare col punto interrogativo (?). Invece di POKE potete scrivere il carattere "P" seguito dal carattere che viene visualizzato premendo contemporaneamente il tasto shift insieme con "O". Tutte le abbreviazioni possibili, lo ripetiamo, sono riportate in una delle appendici di qualsiasi manuale Commodore.

Nel caso sbagliate a digitare i microlistati che superano, in lunghezza, gli ottanta caratteri (SYNTAX ERROR o altri tipi di errore), è necessario, per sicurezza, ribatterli per intero e non apportare modifiche alla riga visualizzata con l'istruzione LIST.

La partecipazione dei lettori è gradita e compensata, in caso di pubblicazione, con materiale della Systems editoriale (libri, fascicoli arretrati, abbonamenti, eccetera).

Inviate le vostre 1 RIGA, purchè nella misura di almeno dieci per volta. Non ci è infatti possibile raggruppare i (moltissimi) micro-programmi che pervengono singolarmente in Redazione.

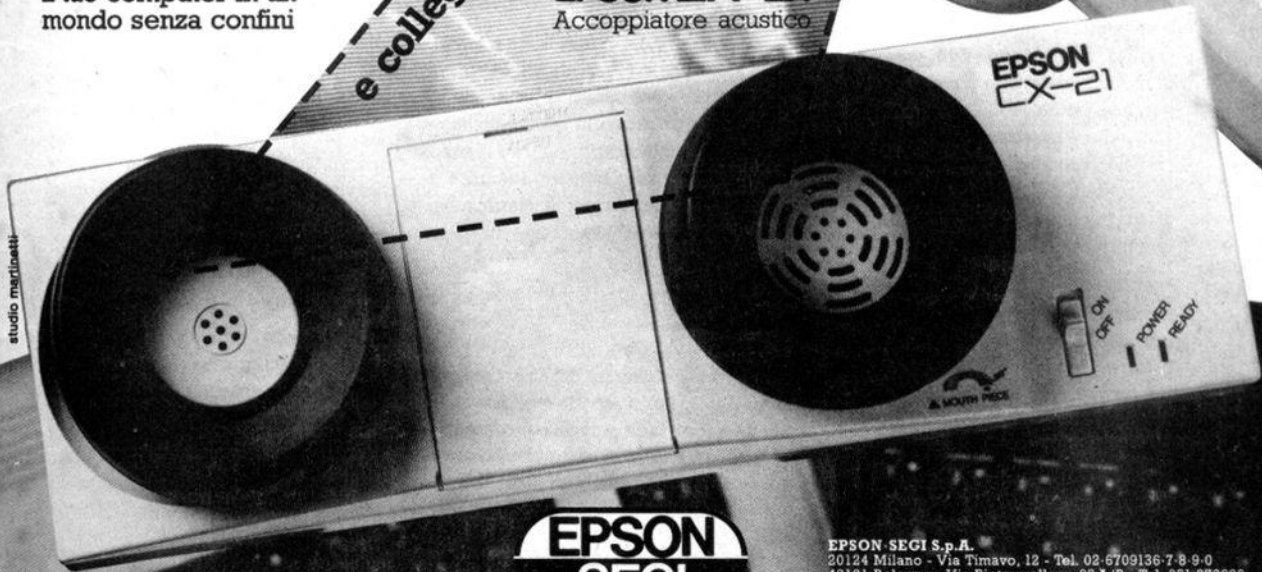
Inviate i vostri lavori su carta, (meglio se su nastro) corredati, ciascuno, di una breve spiegazione sulla funzione che compiono, proprio come li vedete pubblicati in queste pagine.

Ricordate di indicare chiaramente nome, cognome, via e città, e indirizzate a:

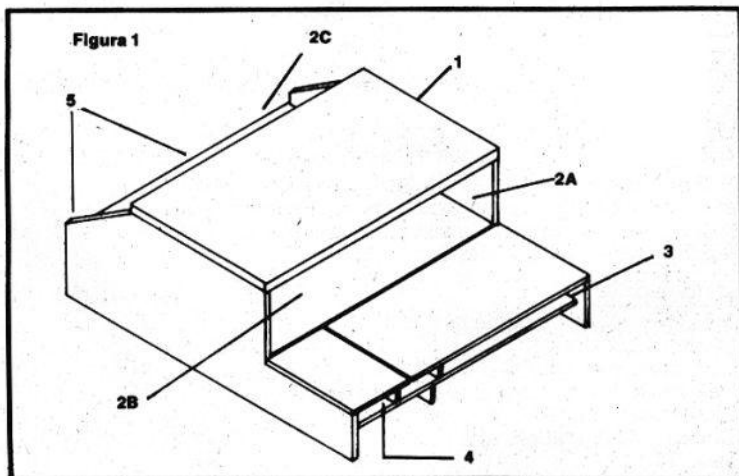
Commodore Computer Club
SYSTEMS EDITORIALE

Rubrica "1 Riga"

Viale Famagosta, 75
20142 MILANO

COMPUTERTELEFONA
I TUOI
PROGRAMMI
CX-21 EPSON è un accessorio che dà un valore enorme al tuo calcolatore, quello dei terminali telecomunicazione.
CX-21 è un modem costruito con le tecnologie più avanzate, (modulatore - demodulatore) capace di convertire i segnali digitali in suoni che possono essere trasmessi con un normale telefono, in tutta sicurezza.
Con CX-21 puoi:
- "telefonare" i programmi direttamente al computer dei tuoi amici che posseggono un CX-21
- comunicare con personal computer di altre marche
- collegarti con le banche dati di tutto il mondo
CX-21 è progettato per adattarsi ai diversi apparecchi telefonici per consentire lo scambio veloce e facile di programmi e dati.
CX-21 è costruito con speciali circuiti integrati CMOS, è piccolo, leggero e molto affidabile.
Caratteristiche
- Funzionamento a batterie ricaricabili (4 ore di autonomia)
- Funzionamento duplex o half duplex
- Velocità di trasmissione fino a 300 bit-sec.
- Interfaccia RS 232
- Dimensioni 297x95x42 mm
- Peso 700 gr.
Inoltre CX-21 può essere collegato a qualsiasi home o personal computer tramite interfaccia RS 232 - C.
e collegati alle banche dati di tutto il mondo
EPSON·SEGI
Garanzia delle Quattro Stagioni
EPSON CX-21,
il tuo computer in un mondo senza confini
EPSON CX-21
Accoppiatore acustico
EPSON
CX-21
ON
OFF
POWER
READY
MOUTH PIECE
studio martinetti
EPSON
SEGI
EPSON·SEGI S.p.A.
20124 Milano - Via Timavo, 12 - Tel. 02·6709136·7·8·9·0
40121 Bologna - Via Pietramellara, 65 A/B - Tel. 051·273686
35128 Padova - Via Pellizzo, 23/9 int. 4/U - Tel. 049·8070870
00199 Roma - Via Asmara, 58 - Tel. 06·8395766

Un po' di ordine sul vostro tavolo

di Alessandro de Simone

Apparecchio TV, computer, registratore, cavi, joystick... Che disordine nella vostra cameretta! Diamo un aspetto più "pulito" alla vostra fantastica centrale operativa.

Il problema dello spazio, nei moderni appartamenti, viene risolto in due fasi successive. I costruttori, dapprima, impongono un numero esiguo di metri quadri s uggerendo implicitamente un rigoroso controllo delle nascite. I genitori, in seguito, costringono i figli, ormai venuti al mondo (e che non possono buttar via perchè è vietato) a tenere tutto in ordine, vale a dire "nascosto" e difficilmente accessibile, in mobili detti eufemisticamente "contenitori", ma che sono in realtà oscuri buchi neri che tutto ingurgitano e divorano.

Il risultato, ovvio, consiste in affannose ricerche di oggetti introvabili oppure, più semplicemente, nella rinuncia a estrarre (nel nostro caso) il computer, il trasformatore e tutti gli accessori d'obbligo, indispensabili per coltivare il proprio hobby.

Il problema, in questo come in altri casi, si risolve attribuendo una veste piacevole al "mostro" computer, alla sua sistemazione stabile sul tavolo della stanza in modo tale che anche i genitori più burberi possano esclamare Ohhhh! alla sua vista e consentirne la permanenza.

Alcuni genitori, o fratellame e sorellame vario che con voi divide la stanza, potrebbero considerare diversamente il progetto presentato in queste pagine e suggerire una ubicazione meno dignitosa. In questo caso siete autorizzati a richiedere un certificato di martirio che potrà conferirvi vantaggi in sede pensionistica.

Il disegno

Come si può notare dai chiari disegni, il mobiletto ha un aspetto spartano e tale scelta è stata fatta proprio per consentire a qualsiasi lettore di costruirne un esemplare ricorrendo a utensili di facile reperibilità. Anche chi non possiede un'attrezzatura per lavorare il legno, non dovrebbe avere difficoltà a rintracciare un falegname in grado di realizzare, per pochi milioni di lire (isole comprese), i semplici pezzi di legno.

Il problema che si è cercato di risolvere è quello di "verticalizzare" la disposizione dei componenti.

Se pensate all'attuale sistemazione "orizzontale" del vostro sistema, potete osservare che occorre lo spazio di un intero tavolino per ospitare il materiale indispensabile: computer, trasformatore, registratore, apparecchio TV, joystick, cassette, oltre all'intrico di cavi vari.

Chi possiede anche un disk drive 1541 e, magari, una stampante, ha inoltre il problema della sistemazione dei dischi e del pacco di carta.

Come è possibile avere un po' d'ordine sulla scrivania e, soprattutto, come risparmiare spazio in modo da utilizzarla

"normalmente" quando il computer è spento?

Abbiamo pensato, l'idea è vecchia, di sfruttare lo spazio dedicato alla TV. Sollevandola, cioè, dal piano della scrivania, si crea automaticamente lo spazio sufficiente per la sistemazione del trasformatore, del registratore, del drive, di accessori vari e, inoltre, si possono occultare gran parte dei cavi che, in verità, sono piuttosto brutti a vedersi.

Considerando la figura 1, ecco come molti di noi, in redazione, utilizzano lo spazio ed i vani del mobiletto in questione:

25
16
19
12
11,4
7,6
A
Spalla
N. pezzi: 2
Spessore: 10 mm.
52
36
C
Piano interno
N. pezzi: 1
Spessore: 10 mm
41
25
B
Piano superiore
N. pezzi: 1
Spessore: 15 mm.
Multistrato
43

D
Reggipiano interno
N. pezzi 2
Spessore: 10 mm.
7
36
E
Piano accessori interno
N. pezzi: 1
Spessore: 4 mm.
14
41
F
Coperchio
accessori
N. pezzi: 1
Spessore: 4 mm.
16
11
G
Coperchio accessori
N. pezzi: 1
Spessore 4 mm.
16
32
Reggipiano
Accessori
N. pezzi: 1
Spessore 10 mm.
H
3,5
10
Divisorio accessori
N. pezzi: 2
Spessore 10 mm.
I
3,7
14

1 - ripiano per il monitor TV;

2 - vano per drive (2-B, a sinistra), per il registratore (2-A, a destra), trasformatore di alimentazione del computer (2-C, dietro al registratore);

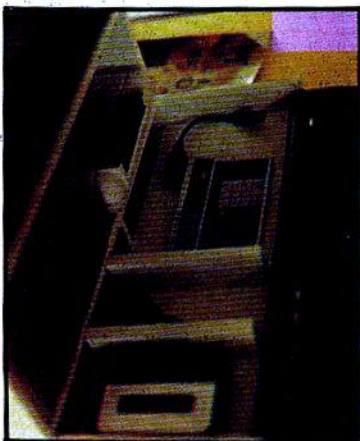
3 - vano accessibile grazie al sollevamento del coperchio "G" (figura 2) in cui sono alloggiate cartucce varie di uso meno frequente (Koala, interfacce, duplicatori di nastro ed altro);

4 - vano (accessibile sollevando il coperchio "F") destinato a ospitare una calcolatrice tascabile (utile mentre si lavora in W/P ed è necessario effettuare calcoli senza ricorrere al computer);

5 - a questo vano (in verità un po' scomodo da raggiungere) si accede sollevando e allontanando dapprima i coperchi "G" ed "F" e poi la mensolina (con l'intero suo contenuto) costituita dai pezzi "E", "I", "H" che, incollati tra loro (figura 2)

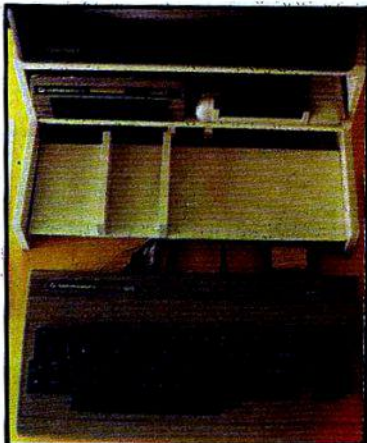
non sono però fissati stabilmente alla struttura, ma semplicemente appoggiati alle costolette "L" poste lateralmente. In fondo a questo vano (5), dunque, trovano posto un joystick, un Koala e le paddle.

I due pezzi "D" servono, oltre che a sostenere il peso del drive, anche a incanalare i cavi del registratore, del drive e della TV che, altrimenti, darebbero fastidio nelle operazioni di sistemazione e rimozione degli accessori.

Al di sotto della mensoletta "E", tra l'altro, vi è spazio sufficiente per la sistemazione stabile, sul retro del calcolatore, di una cartuccia (sulla destra) e di un'interfaccia o altro accessorio da tenere costantemente inserito nella porta

utente (a sinistra).

Per ciò che riguarda gli spessori del materiale (i nostri esemplari sono in compensato) c'è da rilevare quello di almeno 15 millimetri per il ripiano destinato alla TV (altrimenti si incurva per il peso). Gli elementi "E", "F", "G", grazie ai minimi sforzi richiesti, possono esser realizzati con lo spessore di appena 4 millimetri. Tutti gli altri elementi vanno realizzati, invece, in 10 millimetri di spessore.

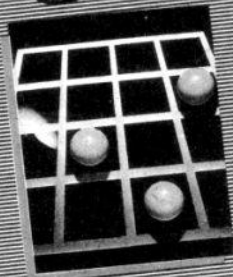
Se giudicate la forma del mobile troppo tozza e se avete gli strumenti idonei a disposizione, potete migliorare il design, ma vi assicuriamo che, una volta realizzato, un paio di semplici "mani" di verni-

ce conferirà un aspetto più che dignitoso al vostro contenitore per computer.

La stabilità del tutto, dimenticavamo di dirlo, è affidata a colla Vinavil e chiodi senza testa.

Avvertenze

Alcuni TV o monitor, sistemati al di sopra o nei pressi di memorie di massa (registratori, drive) possono creare fastidiose interferenze con la trasmissione dei dati. Allo scopo di evitarle, abbiamo sistemato, nella parte inferiore del ripiano 1 (in posizione, cioè, invisibile all'esterno), un sottile foglio di alluminio del tipo impiegato normalmente nella conservazione dei cibi (Domopack). E' sufficiente questo accorgimento per evitare spiacevoli sorprese.

Per il resto, che dirvi? L'idea è lanciata. Se il modello non vi piace, o se avete altre esigenze, modificate il disegno fino a realizzare qualcosa di utile. I più arditi possono, ad esempio, pensare ad inserire anche la stampante....

Si rigrazia Daniele Galimberti, studente dell'Istituto Professionale per il Mobile e l'Arredamento di Lissone, per il disegno esecutivo e la realizzazione del prototipo presentato in queste pagine.

La... moltiplicazione delle tracce e dei settori

di Alessandro de Simone

...ed io vi dico che, se avete fede, raddoppierete la capacità dei vostri dischi. E il povero vedrà raddoppiare le sue sostanze, lo squattrinato risparmierà e lo smemorato ricorderà... (dal Libro delle Memorie - Tomo 3, salmo 8, versetti 15 e segg.)

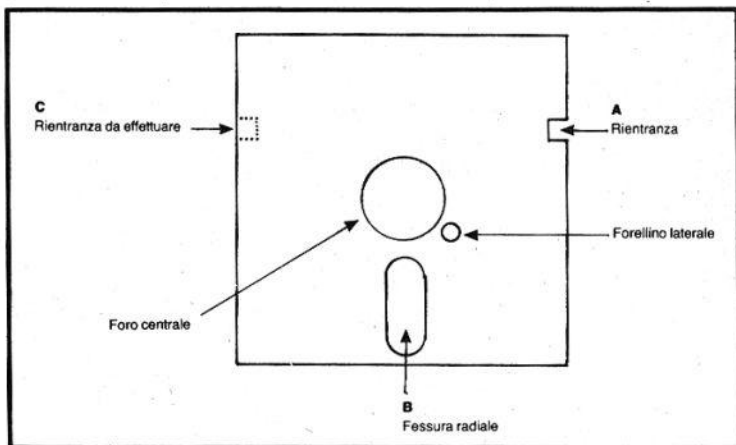
Pochi sanno che nella fabbricazione in generale non sempre si seguono catene di produzione differenziate tra loro. Siamo infatti abituati a vedere in Tv documentari sulla produzione di automobili in cui si parla di catene di montaggio ben distinte per i diversi modelli.

Nel campo dell'elettronica e dei dischi in particolare, molto spesso vi è una sola catena di produzione per diversi prodotti. Nel realizzare i dischi, infatti, il fabbricante "cerca" di produrre solo dischi a doppia faccia-doppia densità, in modo da realizzare un utile più elevato. I macchinari utilizzati, però, pur se sofisticati, non riescono a fornire prodotti perfetti e, in una certa percentuale, si ottengono i cosiddetti scarti di produzione.

Un prodotto viene però definito di scarto solo quando è inutilizzabile per gli scopi prefissati. Spieghiamoci meglio: un dischetto, per essere posto in commercio con l'etichetta "doppia faccia" deve possedere alcune caratteristiche di un certo livello e comunque in conformità a convenzioni internazionali, le cosiddette specifiche. Quando i dischetti escono dalla catena di produzione, vengono testati, uno per uno, al 100% sia da una parte che dall'altra; ciò significa che vengono effettuate prove di scrittura e di lettura per esaminarne il comportamento. Dopo la prova, in base ai risultati conseguiti, si stabilisce se il disco rispetta le specifiche su entrambe le facce oppure no.

In caso affermativo viene posto in commercio con l'etichetta "doppia faccia" altrimenti, se le specifiche sono soddisfatte su un solo lato, vengono confezionati come "singola faccia".

In questo modo, tenendo attiva una sola linea di produzione, si ottiene un risparmio rispetto alla produzione differenziata che richiederebbe più macchinari in funzione contemporaneamente.

Ma se in un dischetto una delle facce non rispetta le specifiche, non vuol dire che entrambe le superfici siano assolutamente inservibili, ma semplicemente che una delle due non ha superato l'esame imposto dalle rigorosissime regole internazionali. In altre parole, la faccia "bocciata" del dischetto è ancora magnetizzabile e, di conseguenza, utilizzabile, pur se con una garanzia di funzionamento ridotta.

La faccia "buona", insomma, è garantita al 100%, mentre quella... "cattiva" lo è al 99,9%. Si tenga presente infatti che le moderne macchine di produzione non scendono mai al di sotto di un'affidabilità che rimane, in ogni caso, molto elevata.

Dalla teoria alla pratica

Anzitutto precisiamo che il trucchetto che stiamo per insegnarvi vale **soltanto** per le unità a dischetto predisposte per minifloppy a singola faccia, come è il caso del drive 1541. I drive che richiedono dischi a doppia faccia, utilizzano di già, come è intuitivo, entrambi i lati del disco.

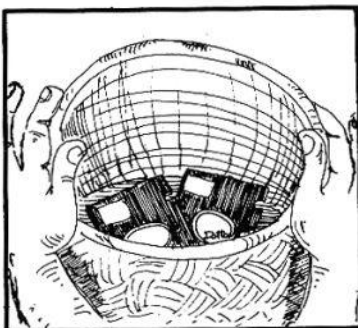
Prendete dunque un dischetto e praticate la rientranza C come indicato in figura. Nell'effettuarla, anche servendosi di comuni forbici, si presti la **massima** attenzione a rispettare le regole che seguono:

- non toccare con le mani la fessura radiale B;
- non piegare il dischetto;
- praticare la rientranza in misura eguale a quella A: esagerando col taglio rischiereste di tagliare anche il disco;
- prendere con precisione le misure prima di effettuare il taglio. Ricordarsi che questo deve essere perfettamente simmetrico a quello già esistente.

Fatto questo, inserite il dischetto, capovolto, nel drive e cioè semplicemente al contrario (destra-sinistra) di come siete abituati a fare. Formattate e... il gioco è fatto. Il dischetto è diventato un "quasi" doppia faccia.

Non si dimentichi però che la nuova faccia recuperata nel modo descritto non è garantita al 100% ed è pertanto prudente depositarvi dati non molto importanti. Nel caso di salvataggio di programmi, ad esempio, si effettui sempre un'operazione di verifica prima di cancellarli dalla memoria del computer.

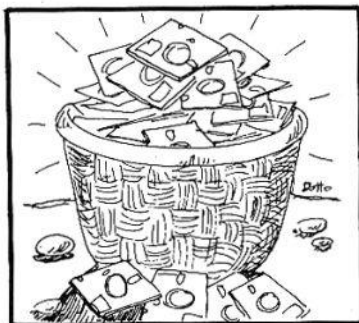
Vi possiamo però assicurare che, al-

meno seguendo la nostra esperienza personale e quella di altri collaboratori e lettori, finora non è andato perso un solo dato, anche perchè sono stati usati dischi di marche di solida tradizione e serietà.

E il forellino laterale?

Qualcuno potrebbe obiettare che, rivoltando il disco, risulta necessario praticare un nuovo forellino, simile a quello già esistente, in modo da ripristinare la simmetria originale.

Ebbene, quel forellino non serve assolutamente a nulla e potete sincerarvene coprendolo con un po' di nastro adesivo: anche se "accecato" in questo modo, il disco funziona egualmente.

A che serve, allora, questo benedetto forellino e perchè viene praticato sui dischi se non viene utilizzato in alcun modo?

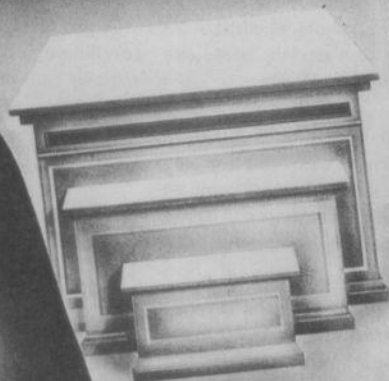
Il motivo risiede nel fatto che il drive 1541 è del tipo S/S (Software Sectored).

Ciò significa che, al momento della formattazione, il drive trascrive, in un punto assolutamente casuale della superficie magnetica, una serie di informazioni che terrà come riferimento sia nel memorizzare dati che nel leggerli successivamente. In altre parole, al momento dell'utilizzo, la testina si posizionerà in un punto ben preciso del raggio e farà girare il disco fino a che non rintraccia il punto di riferimento precedentemente generato.

Altri drive, di altre marche, sono invece del tipo H/S (Hardware Sectored). Questi, come è intuibile, utilizzano, come riferimento, la zona magnetica individuata dal forellino. Questo, nella rotazione del disco, fa passare un raggio di luce che, intercettato da una cellula fotoelettrica sistemata sul lato opposto, comunica al sistema operativo del disco l'avvenuto "incontro" con la zona di riferimento.

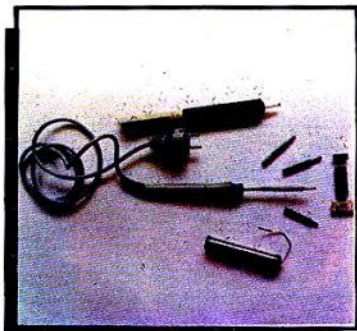
Il forellino, in conclusione, consente di porre in commercio lo stesso disco sia per sistemi S/S che per sistemi H/S. In alcuni casi, inoltre, viene utilizzato in fabbrica per sveltire i processi di controllo di qualità.

I dischi a doppia faccia

Il pericolo di avere la "seconda" faccia di qualità dubbia non viene corso da chi possiede dischi venduti con l'etichetta "doppia faccia". In questo caso è evidente che la garanzia di funzionamento è estesa ad ambo i lati del minifloppy. Ricordiamo, ad ogni buon conto, che per garanzia non intendiamo quella eventuale della casa costruttrice (che, anzi, decade dato che manomettiamo il prodotto), garanzia nel senso di sicurezza di funzionamento.

A proposito, che fine fanno i dischetti che non superano la prova su nessuna delle due facce? Alcuni dicono che vengono buttati via, altri, più maligni, asseriscono (ma sarà vero?) che vengono posti in commercio con marche sconosciute o addirittura senza nome oppure dati in omaggio... Meditate, gente, meditate.

Un pulsante di Reset

di Alessandro de Simone

Il vostro primo incontro col saldatore nella realizzazione di un semplice accessorio indispensabile in un personal computer.

I circuiti di un qualsiasi apparecchio elettronico sono collegati tra loro per mezzo di cavi oppure piste in rame dei circuiti stampati. Il costante contatto el ettrico tra le centinaia di componenti (piedini di circuiti integrati, terminali di resistenze, consensatori eccetera) è affidato a due sistemi.

Il primo, ricorrendo all'inserimento forzato dei piedini in appositi alloggiamenti, è di tipo, appunto, meccanico. Il vantaggio consiste nella rapidità di inserimento ed estrazione del componente.

Il secondo, che offre la massima garanzia di collegamento, si effettua mediante la fusione di una lega di stagno che, fatta cadere in minima quantità (goccia) tra le estremità di due componenti ramati, li preserva da ossidazioni ed interruzioni accidentali.

Lo svantaggio di una saldatura consiste in una più difficoltosa opera di distacco nel caso risulti necessario rimuovere un componente difettoso, per la sua sostituzione.

Il saldatore

Il primo utensile che ogni hobbista deve possedere è, appunto, il saldatore. Dato che verrà usato soprattutto per collegare stabilmente componenti elettronici di dimensioni ridotte, sarà opportuno procurarsene uno di basso wattaggio (circa 16 watt), specifico per circuiti integrati.

Saldatori di maggiore potenza (40 watt), infatti, possono procurare danni irreparabili a componenti delicati. Nel dubbio, rinunciate all'acquisto e, in ogni caso, non cercate di risparmiare qualche migliaio di lire nel comprare un saldatore più potente, più economico ma indatto allo scopo.

Insieme al saldatore, procuratevi stagno di buona qualità (evitate risparmi illusori) e, crepi l'avarizia, uno spellafili ed un dissaldatore ad aspirazione. Il primo eviterà la rottura dei vostri incisivi nei frequenti tentativi di utilizzare i denti allo scopo di "spellare" le guaine isolanti dei cavi elettrici. Il secondo (che funziona aspirando lo stagno reso liquido per la fusione) sarà preziosissimo nel dissaldare componenti da sostituire, o collegati fra loro in modo errato.

Tenete conto che la temperatura di fusione dello stagno supera di gran lunga quella dell'acqua calda. Evitate accuratamente di farvi gocciolare addosso rivoli di stagno fuso e, soprattutto, non fate cadere gocce argentee e fumant piano del tavolo del salotto.

Quando il saldatore è acceso, il suo aspetto è subdolamente tranquillo: non emette alcun suono nè si accende una spia.

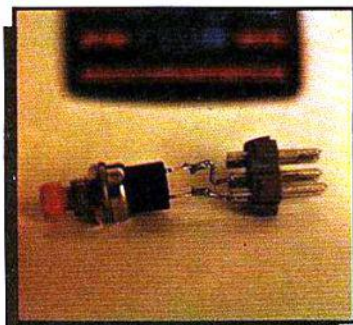
Attenti però a non prenderlo inavvertitamente dal lato metallico. Ùn urlo acuto segnalerà, in quest'ultimo caso, che il saldatore era in funzione.

In seguito alle primissime ustioni, guaribili in pochi giorni, avrete accumulato una sufficiente esperienza su come lavorare e, soprattutto, come **non** lavorare col saldatore.

Un po' di teoria

Quando un calcolatore viene acceso, il suo microprocassore "si sveglia" per la tensione applicata e, grazie ad una particolare logica elettronica predisposta, esegue **sempre** un particolare programma detto in gergo di "inizializzazione".

L'utente del Commodore 64, in particolare, può rendersi conto dell'effetto di tale procedura osservando il solito messaggio:

```
** COMMODORE 64 BASIC V2 **
64 RAM SYSTEM 38911 BASIC
BYTES FREE
```

o altri analoghi per gli altri computer, che appaiono al momento dell'accensione.

Non tutti sanno che è possibile riprodurre l'inizializzazione descritta senza spegnere e riaccendere il computer ma simulando, via Hardware, lo spegnimento e la riaccensione.

Il trucco consiste nel mettere a massa (detta anche ground) la pista ramata del piedino reset del microprocessore.

In altre parole si tratta di realizzare un corto-circuito tra il pin (=piedino) di Reset del cervello del vostro computer, e un qualsiasi punto di massa.

Sembrerebbe necessario aprire il contenitore ed effettuare pericolose quanto difficili operazioni chirurgiche. Fortunata mente è possibile aggirare l'ostacolo ricorrendo, esternamente, al connettore utilizzato di norma per collegarsi con periferiche seriali, come il drive, la stampante, il plottter, eccetera.

Su tale connettore, infatti, sono presenti sia una "presa" di massa, sia una presa di reset che, all'interno del computer, sono in diretto contatto elettrico con le piste ramate che ci interessano.

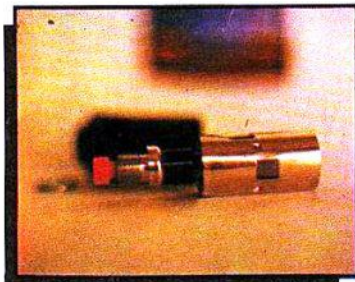
Purtroppo non tutti gli esemplari Commodore sono collegati allo stesso modo e, in questi casi, mettere in corto circuito i due pin menzionati non sortisce alcun effetto.

Per controllare se il vostro computer consente il reset hardware, effettuate la prova seguente:

- procuratevi una graffetta, di quelle usate normalmente negli uffici, e "apritela" in modo da ottenere un lungo filo metallico.
- Piegatelo ad "U" e, prestando la **massima** attenzione, inserite per un solo breve istante le sue estremità nei forellini del connettore seriale contrassegnati da RST (=reset) e GND (=ground). Nell'appendice "I" del manuale del vostro computer è riportato lo schema del connettore seriale con le relative indicazioni utili a rintracciare i pin in questione. Prima di effettuare questa semplice operazione, digitate qualcosa sulio schermo. Subito dopo il distacco della graffetta (che deve, ripetiamo, provocare il contatto tra i due pin per un solo breve istante) dovreste notare lo stesso effetto che si ottiene con le SYS di reset, vale a dire la cancellazione dello schermo e l'apparizione del consueto messaggio che si ottiene, tra l'altro, anche al momento dell'accensione del calcolatore.

Se la prova descritta ha fornito esito positivo, potete costruirvi il comodo accessorio descritto in queste pagine. In caso contrario, non sarà semplice realizzare il pulsante di reset.

Costruzione del pulsante

Procuratevi, da un negozio specializzato, uno spinotto (maschio) a sei pin (compreso quello centrale, che è il più importante). Poichè vi sono molti tipi di spinotti a sei pin, incompatibili tra loro, è bene andare in negozio col cavetto del drive o della stampante (se li possedete) come campione oppure col libretto di istruzioni in modo da mostrare il disegno. I più prudenti potranno portare con sè il computer in modo da evitare equivoci.

Acquistate anche un minipulsante (e non un interruttore) del tipo normalmente aperto. Ciò significa che i due terminali del pulsante vengono in contatto tra loro solo al momento della pressione del tasto. Normalmente non c'è contatto e non vi deve essere, altrimenti il computer sarebbe in stato permanente di reset!

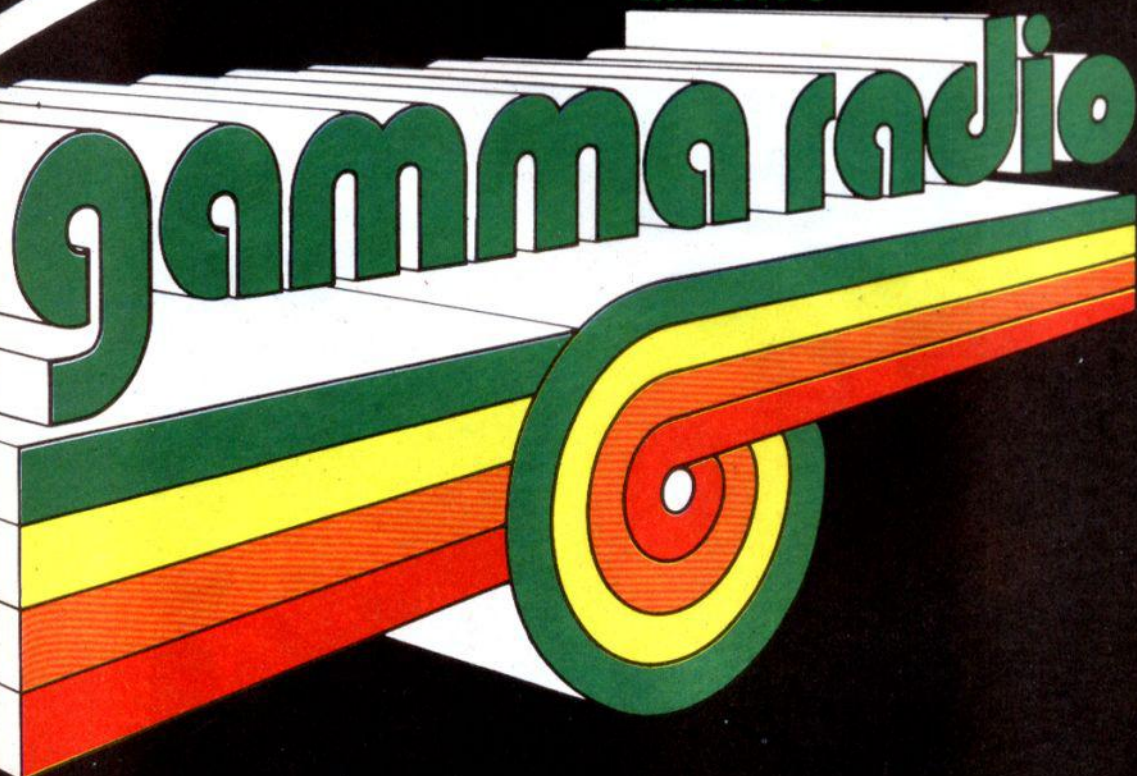
Attenti al pin

Tornati a casa, smontate lo spinotto e saldata le sue estremità ai pin Reset e Ground facendo bene attenzione a non provocare corti circuiti con i pin adiacenti. Per individuare il pin di Reset non c'è problema: è quello centrale. Il pin di massa, invece, può facilmente esser confuso a seconda di come si "guarda" il connettore, se da sinistra o da destra.

Effettuate le due saldature, inserite lo spinotto nella presa seriale e accendete il computer. Digitate qualcosa sul video e premete (per un breve istante) il pulsante appena costruito. Se vi scoppia tutto in faccia contattate prontamente un medico. In caso di incendio è bene però rivolgersi ai pompieri.

Se tutto invece funziona a dovere siate contenti: avete realizzato un accessorio hardware di notevole utilità che aprirà le porte del futuro al cammino dell'uomo, padrone del mondo.

(Il film si chiude in dissolvenza sul vostro volto, illuminato dal sole, che guarda verso il cielo. Dalla bocca fuoriesce, come una sigaretta, lo spinotto, spento. Una musica di sottofondo, prodotta da un'orchestra di venti elementi, supporta un coro che canta "Un bel dì vedremo").

24 ORE SU 24 DI MUSICA IN STEREOFONIA CON

CONCESSIONARIA PER LA PUBBLICITÀ DI MILANO

RADIANT S.P.A.

CONCESSIONARIA PER LA PUBBLICITÀ DEL CIRCUITO

gamma color italia S.r.l.

PALAZZO CANOVA CENTRO DIREZIONALE MILANO 2 - 20090 SEGRATE (MI)
TEL. 02/2155714 - 2155726 - 2155734

LOMBARDIA	
Milano	95.9-92.8-97.1
Bergamo	99.3
Brescia	92-92.7
Como	97.1
Cremona	99.3
Pavia	95.9-97.1
Varese	94.9

LIGURIA	
Genova	96.25
La Spezia	98.7

EMILIA ROMAGNA	
Bologna	88.7
Piacenza	97.1

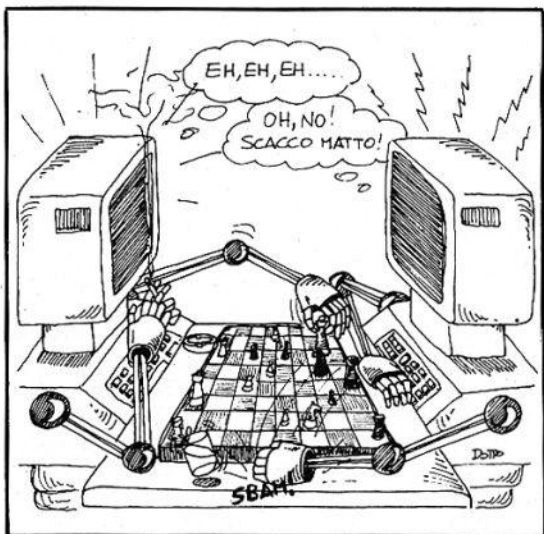
PIEMONTE/VAL D'AOSTA	
Alessandria	104.3
Cuneo	90.6-97.6
Novara	97.1
Aosta	91.8-92

TOSCANA	
Firenze	97.6-104.4
Livorno	98.2-97.3-100.6
Massa C.	98.7
Pistoia	97.6-104.4
Pisa	97.3
Lucca	97.3

LAZIO	
Roma	99.5

C 64 e Plus 4 a confronto

di Michele Maggi

Se sto per comprare uno dei due, quali problemi si presentano e, soprattutto, quali soluzioni possono offrire i due noti computer?

Sulla linea di partenza l'ormai famosissimo e diffusissimo C64 affiancato dal più recente Plus/4: si tratta di due prodotti per cui questa specie di gara tende non a identificare il "migliore", ma a mettere in luce le caratteristiche più salienti dei due modelli.

Notiamo prima di tutto che si tratta di due home computer rivolti a due tipi di utenti: da un lato il C64 richiama un pubblico eterogeneo e giovane, mentre il "target" a cui è rivolto il Plus/4 è un po' meno giovane ma più interessato ad utilizzare realmente il computer ad un livello, tra l'altro, professionale.

Vediamone i motivi.

Prima di tutto, il fatto che il C64 sia in grado di creare e controllare ben otto sprites indica questo prodotto come un fertilissimo campo per la creazione di videogames, per cui è, e sarà, ancora per diverso tempo un'ottima consolle per videogiochi.

A parte l'aspetto ludico, esiste anche l'aspetto "professionale" (nel caso del 64, "studentesco") che invoglia ad usarlo anche in campi più impegnativi.

Il basic 2.0 del 64 è tipica e permette di creare senza troppe difficoltà numerosi programmi di vario genere: matematici, statistici e così via.

Un problema notevole può, però, sorgere quando si ha bisogno di disegnare in alta risoluzione. Mentre, infatti, il Plus/4 mette a disposizione istruzioni specifiche che permettono un facile e immediato uso della pagina grafica, il C64 richiede procedure più complesse e lente e, comunque, non offre tutte le possibilità grafiche offerte dal Plus/4.

Se però consideriamo il Plus/4 superiore al C64 per ciò che riguarda la semplicitàdi gestione della grafica (sprites a parte) dobbiamo d'altra parte spezzare una lancia a favore di quest'ultimo per ciò che riguarda la generazione di suoni e di musica.

Il C64, infatti, può essere considerato un vero sintetizzatore musicale, anche se di livello non professionale.

Per ciò che riguarda ancora la musica esistono in commercio numerosissimi prodotti software e hardware che permettono di utilizzare il computer in modo davvero interessante e creativo.

Dal punto di vista "sonoro", invece, il Plus 4 è decisamente inferiore, anche se l'utilizzo dei suoni ècomandabile da basic con due specifiche istruzioni: VOL e SOUND. Questa considerazione non fa che sottolineare ancora una volta che il Plus/4 è destinato a un pubblico formato soprattutto da persone che hanno necessità, per motivi di lavoro, di un buon computer a basso prezzo.

Ma veniamo alla caratteristica più importante del Plus/4 che lo rende davvero unico nel suo genere.

Il software integrato

Gran parte della pubblicità fatta per il Plus/4, lo descrive come il computer "con quattro programmi dentro". Vediamo di analizzare questi quattro (o meglio 3+1) programmi. Si tratta di un Word Processor, di uno Spreadsheet, di un Data Base e di un programma di grafica.

Questi programmi sono perfettamente integrati e ciò significa, ad esempio, che senza lunghi e macchinosi procedimenti qualsiasi foglio elettronico, o parte di esso, può essere richiamato dal Word Processor in modo da essere successivamente editato, etichettato o modificato.

Questo fatto rende il Plus/4 sicuramente un prodotto unico nel suo genere

e indispensabile per chi voglia utilizzare un computer subito e al meglio senza dover tassativamente imparare un linguaggio di programmazione.

Il Word Processor, lo Spreadsheet e il Data Base sono tre programmi veri e propri mentre il programma di grafica è semplicemente un'estensione del Word Processor; la scelta del tipo di grafica da utilizzare è caduta sulla grafica in bassa risoluzione essenzialmente per due motivi:

- se si fosse usata l'alta risoluzione non sarebbe stato poi possibile stampare un grafico con stampanti non Commodore o comunque non di tipo grafico;
- non sarebbe stato possibile realizzare la fusione tra un grafico e un documento del Word Processor.

Tutto ciò comunque rappresenta un buon compromesso tra potenza e facilità d'uso.

Per i patiti dell'alta risoluzione si segnala che mediante le istruzioni grafiche è comunque possibile creare facilmente programmi grafici di grande utilità.

Un'ultima considerazione prima di passare all'analisi del software integrato: il C64 dispone all'accensione di circa 39k di memoria mentre il Plus/4 di più di 60K.

Il word processor

Si tratta di un modesto Word Processor che consente una qualità di lavoro accettabile (quest'articolo è stato appunto scritto con il WP in questione), abbinata ad una notevole facilità d'uso.

Non si vuole in questa sede analizzare singolarmente ogni comando del WP ma solo valutarne la potenza e la versatilità.

Se lo paragoniamo ad un diffuso WP per il C64 come l'Easy Script, notiamo che a fronte di una maggiore immediatezza il WP del Plus/4 è più limitato, ma non tanto da compromettere la qualità dei documenti che possono venire elaborati. E', infatti, possibile stabilire la lunghezza della pagina, le dimensioni del formato del foglio, la stampa in reverse e ingrandito. Esistono insomma in questo WP tutte le qualità che ne fanno un prodotto valido.

Una limitazione è causata non solo dalla lunghezza massima consentita (appena 99 righe), ma anche dal fatto che non è possibile visualizzare sullo schermo il documento nel modo in cui sarà stampato su carta.

Lo spreadsheet

Anche per lo spreadsheet valgono le stesse considerazioni fatte per il WP. Infatti esistono in commercio, per il C64, numerosi spreadsheets sicuramente più completi. La limitazione derivante dal fatto che le dimensioni della cella sono fisse, mentre in altri spreadsheets è possibile definirle prima di utilizzalo, può essere seccante per chi è abituato diversamente.

Il punto di forza di questo foglio elettronico è dato dal fatto che è integrabile con il WP nel senso che, come si diceva prima, sarà possibile, con un particolare comando (BLKMAP), trasportare i contenuti delle celle dello Spreadsheet nel WP in modo da aggiungere dei commenti.

Il data base

Con questo Data Base è possibile definire fino a 999 schede, ognuna delle quali estraibile, modificabile e rivedibile con semplici comandi.

Ogni record può contenere fino a 17 campi, ognuno dei quali lungo fino a 38 caratteri.

E' inutile ricordare che anche il Data Base è integrabile con il WP per un utilizzo che prevede, tra l'altro, l'estrazione di informazioni specifiche dal file per associarle ad una lettera o ad un invito o ad un altro qualsiasi documento.

Il confronto

Vediamo ora se, e come, è possibile ottenere dal C64 prestazioni analoghe (!) al Plus/4.

Innanzittutto la versione base del Plus/4 è composta da: PLUS/4, disk drive e stampante. Le due periferiche sono indispensabili per sfruttare le possibilità della macchina.

In termini monetari tutto ciò significa:

Plus/4	449.000
Drive 1541	630.000
stampante MPS 801	290.000
TOTALE	1.369.000

Vediamo ora, oltre alle periferiche in questione, quanto potrebbe incidere il costo del software per il C64 in modo da renderlo il più possibile simile (o meglio, nettamente superiore) al Plus/4.

Commodore 64	399.000
Drive 1541	630.000
Stampante MPS 801	290.000
Easy Script	31.000
The Manager	31.000
Calc Result	65.000
Simons' Basic	85.000
Assembler	31.000
TOTALE	1.562.000

Ricorderemo che l'Easy Script è un Word Processor, The Manager è un Data Base, Calc Result è uno Spreadsheet; Simons' Basic è un programma che aggiunge circa 100 comandi, soprattutto grafici, mentre Assembler è un programma per programmare in L.M. cosa che col Plus/4 è ottenibile mediante il Monitor incorporato. In sostanza la differenza in termini monetari si riduce a poco meno di 200.000 lire.

Che fare?

La risposta dipende chiaramente dalle esigenze di ognuno; per ridurre la spesa per il software del C64 si può sostituire il Simons' Basic con le Routines grafiche di D. Toma (C.C.C. 14), e rinunciare temporaneamente all'Assembler o al software che meno interessa in modo da far pareggiare i totali.

Resta però il fatto che il software integrato del Plus/4 offre l'enorme comodità di averlo disponibile subito, all'accensione del computer.

A voi la scelta!

Firma i tuoi listati

di Alessandro de Simone

Impedire le copie abusive dei programmi è impossibile. Ecco, però, un sistema per aiutare un giudice, chiamato in causa per riconoscere la paternità del vostro software posto illegalmente in vendita, a darvi ragione.

Sul numero precedente di Commodore Computer Club abbiamo iniziato il discorso sulle protezioni proponendo tre listati idonei a render la vita difficile ai copiatori abusivi.

Avevamo anche promesso di continuare la chiacchierata suggerendo alcune idee da elaborare e da inserire opportunamente nei programmi che avete intenzione di vendere a terzi.

Dovrebbe esser chiaro che è matematicamente impossibile realizzare un sistema software (o anche hardware) che impedisca le copie abusive. Se infatti esistesse, e fosse economicamente conveniente, sarebbe adottato universalmente e non si assisterebbe al fenomeno della pirateria.

Recentemente alcuni giudici, chiamati a dipanare problemi sui diritti d'autore relativi a programmi, hanno dato ragione agli autori di validi videogiochi. Copie abusive di questo software, infatti, erano state poste in vendita a prezzi bassissimi e senza l'autorizzazione dei realizzatori.

I giudici hanno potuto dare ragione alla "parte lesa", grazie anche alla facilo-

neria dei copiatori, che non si erano nemmeno presi la briga di togliere le videate iniziali in cui compariva in bella mostra il nome della software house.

Sarebbe andata allo stesso modo se i copiatori avessero avuto l'accortezza di cancellare almeno quel nome?

Sarebbe stato sufficiente, per i giudici, esaminare i due videogame (originale e non) e stabilire quale dei due fosse copia dell'altro basandosi solo sulla data di apparizione sul mercato dei due prodotti?

Non è possibile, ovviamente, basarsi su ipotesi ma, di certo, un giudice è facilitato nelle sue decisioni se il programma esaminato contiene un "qualcosa" che possa essere considerato come firma dell'autore.

Le firme più utilizzate

Avrete certamente notato che, in molti videogiochi, compaiono messaggi e scritte varie facendo ricorso a caratteri ridefiniti e non utilizzando i caratteri standard.

Ciò è dovuto anche a motivi estetici ma, soprattutto, per impedire di rintracciarli e cancellarli. Uno sprotettore, infatti, spesso non è in grado (o non ha voglia) di individuare e modificare la parte di programma contenente i caratteri ridefiniti o anche gli sprite: si limita a eliminare la protezione originale, e a diffondere sul mercato copie abusive, sperando che nessuno protesti.

Ecco dunque che la superficialità degli sprotettori, da una parte, e la "furbizia" (o è stato un caso?) da parte delle software house e l'intelligenza dei giudici dall'altra, hanno contribuito a stabilire una regola ragionevolmente valida:

Se in un programma posto in commercio è rintracciabile una "firma", risulta molto semplice stabilire il vero autore del software.

Un esempio pratico

E' ovvio che è compito dell'autore inserire la propria "firma" all'interno del programma e fare in modo che la sua presenza sfugga al controllo di un eventuale malintenzionato.

Per uscire dal generico e suggerire al lettore un modo per "firmare" i propri listati, daremo qui di seguito un semplicissimo esempio, applicabile anche da un principiante. Rimane sottinteso che è compito dell'autore di software portare avanti il discorso ed effettuare proprie ricerche in modo da ottenere analoghi risultati.

Supponiamo che il primo listato di queste pagine sia il programma, scritto in Basic, che si vuol firmare.

E' ovvio che è un listato piuttosto stupido, che non merita certo alcuna protezione. Servirà solo da esempio per illustrare meglio le righe da aggiungere in modo che visualizzino la vostra "firma".

Il programma chiede all'utilizzatore di digitare il risultato di una somma di due numeri (X e Y) di due cifre ciascuno. Comparirà la scritta "ESATTO" se indoviniate e "ERRATO" in caso contrario.

Ricordiamo, prima di proseguire, che la vostra firma non deve risultare sempre visibile, altrimenti lo sprotettore può esser invogliato a rintracciare la parte del listato in cui è presente e cancellarla. La vostra firma, insomma, dovrà apparire solo "su comando" e, ovviamente, solo voi dovete conoscere il modo per farla apparire su video (o stampante o disco).

Se in presenza del giudice, in un'eventuale vertenza legale, voi stessi fate comparire il vostro nome "estraendolo" dal programma posto in vendita abusivamente, nessuno potrà negare che il programma sia vostro!

Il secondo programma (Firma indelebile) rappresenta lo stesso listato di prima con alcune righe in più.

Esaminiamole:

100 - La variabile del tempo TI$ viene azzerata e da questo istante l'orologio del vostro Commodore inizierà il conteggio del tempo, incrementando periodicamente il valore di TI$.

150 - Ogni volta che viene digitata una risposta, viene associata alla variabile "Z" il valore del tempo fino allora trascorso. Se, ad esempio, sono trascorsi sei minuti e 25 secondi, "Z" varrà 625.

160 - In questa riga è nascosta la chiave di accesso alla vostra firma: per come è strutturato il programma non capiterà mai una domanda che richieda una risposta negativa e, in tal modo, si evita che venga visualizzato il vostro nome per cause fortuite. Inoltre (notare l'istruzione IF Z>30 AND Z<35) è possibile saltare alla riga 190 solo se si digita un valore negativo nel ristretto intervallo di tempo (cinque secondi) compreso tra i 30 e 35 secondi dalla partenza del programma.

190 - Una volta che il programma salta a questa riga, compare su video il messaggio di copyright e, in seguito, prosegue normalmente.

In altre parole: è possibile render visibile la vostra firma solo se si digita, come risposta, un valore negativo nell'intervallo di tempo citato. In tutti gli altri casi il programma, pur funzionando perfettamente, non prevede la visualizzazione della firma. E' molto improbabile che, per puro caso, lo sprotettore, o chi non conosce l'indirizzo del salto segreto, riesca inavvertitamente ad attivare il messaggio presente dalla riga 190 alla 210.

Complichiamo la ricerca

E' fin troppo facile obiettare che, listando il programma, balza all'occhio immediatamente il segmento di listato esaminato. Dobbiamo però ricordare che questo può sfuggire in un programma molto più lungo, come un gestionale, che occupa spesso moltissime pagine. Sono questi, infatti, i programmi che val la pena di proteggere dato che, come è noto, è possibile richiedere (e ottenere) anche diversi milioni di lire per un package professionale personalizzato sulle esigenze del cliente.

Non dimentichiamo, inoltre, che, oltre alla "firma" sarà cura dell'autore di software aggiungere altre protezioni come quelle antilist, anti back up e così via.

Il terzo programma di queste pagine non è altro che quello appena visto, con la differenza che la firma, invece che sotto forma di una stringa alfanumerica, viene inserita mediante i DATA di riga 77. Il messaggio che compare, lungo in questo caso solo 13 caratteri (corrispondenti ciascuno ad un valore ASCII) è più difficile da individuare, specie se, come già detto, questi DATA si confondono con altri utilizzati nel listato.

Altri suggerimenti

Il lettore, ne siamo sicuri, sarà certamente in grado di far apparire la propria firma se, ad esempio, si premeno contemporaneamente il tasto Fire di un joystick e il tasto Shift. Oppure se una penna ottica (non prevista, ovviamente, nel programma) tocca in un certo istante il bordo del video o altre combinazioni "starne" come il non toccare nessun tasto per due minuti dal momento del RUN.

Chi, poi, è esperto in linguaggio macchina non ha certo bisogno di suggerimenti per sviluppare le proprie idee.

```
 10 REM  LISTATO DA
 20 REM  PROTEGGERE
 40 :
110 PRINT"QUANTO FA"
120 X=INT(RND(0)*100)
130 Y=INT(RND(0)*100)
140 PRINTX"+"Y;
150 INPUT  A
170 IF A<>X+Y THEN PRINT"ERRATO"
    :GOTO 110
180 PRINT"ESATTO":GOTO 110
```

```
 10 REM  ESEMPIO DI
 20 REM  FIRMA INDELEBILE
 30 REM  IN UN LISTATO
 40 :
100 TI$="000000"
110 PRINT"QUANTO FA"
120 X=INT(RND(0)*100)
130 Y=INT(RND(0)*100)
140 PRINTX"+"Y;
150 INPUT A:Z=VAL(TI$)
160 IF Z>30 AND Z<35 AND A<0 THEN 190
170 IF A<>X+Y THEN PRINT"ERRATO":GOTO 110
180 PRINT"ESATTO":GOTO 110
190 PRINT"QUESTO PROGRAMMA"
200 PRINT"E' STATO SCRITTO"
210 PRINT"DA MARIO BIANCHI"
220 GOTO 110
```

```
100 REM  ESEMPIO DI
110 REM  FIRMA INVISIBILE
120 REM  IN UN LISTATO
130 :
140 TI$="000000"
150 PRINT"QUANTO FA"
160 X=INT(RND(0)*100)
170 Y=INT(RND(0)*100)
180 PRINTX"+"Y;
190 INPUT A:Z=VAL(TI$)
200 IF Z>30 AND Z<35 AND A<0 THEN 230
210 IF A<>X+Y THEN PRINT"ERRATO"
    :GOTO 150
220 PRINT"ESATTO":GOTO 150
230 PRINT:FOR I=1 TO 13
240 READ A:PRINTCHR$(A);:NEXT
250 PRINT:GOTO 150
260 DATA 77,65,82,73,79,32,66,73
    ,65,78,67,72,73
```

PRESENTA

Software Club

20 min. Computer Film

C64/C128

Cover
Buddha
Graphic routine
Computer film
CapTron
Merge

C16/+4

Cover
Data Base
Invaders

MSX

Pinguin
Lunar lander

VC20

Cover
War game
Distruction
Invaders

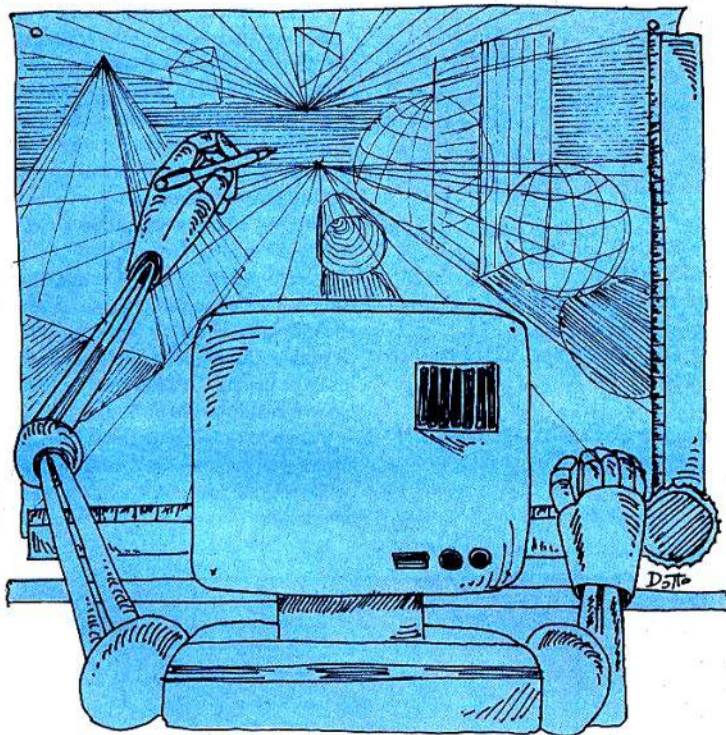
Spectrum

Dancing
Modem
Memorandum
Odissea

In edicola

Commodore Club # 9
Dir. Resp. A. Ronchetti
Edizioni Systems Editoriale Srl - V.le Famagosta 75 - 20142 Milano - Reg. Trib. Mi. n. 104 del 25/2/84 - Distr. MePe

La tua prima volta con la grafica del C 64

di Pasquale D'Andreti

Tutto (o quasi) quello che avrete voluto sapere sulla gestione grafica del popolare computer e che nessuno vi ha finora spiegato, senza arrossire...

In questo articolo tratteremo alcune delle possibilità offerte dal C-64 nel campo delle opzioni grafiche, dal modo testo al modo alta risoluzione, di cui l'utente può usufruire.

Purtroppo il BASIC 2.0 fornito sulle ROM in dotazione al computer non comprende comandi specifici per l'attivazione dei vari modi grafici. Dovremo pertanto ricorrere alle istruzioni PEEK e POKE per modificare alcuni registri del circuito integrato 6567 (chiamato anche VIC II), cui spetta il compito di gestire le immagini che osserviamo sul video. Per comprendere meglio ciò che verrà spiegato in seguito, consigliamo ai lettori di dare un'occhiata alla figura 1 e di leggere con attenzione il riquadro dedicato agli operatori logici e alla rappresentazione binaria di un byte.

Il modo testo

Ciò che leggiamo sul video non è altro che una particolare interpretazione che il circuito elettronico 6567 fa della memoria video e, più precisamente, di due zone di memoria collocate dal byte 1024 a quello 2023 e da 55296 a 56295.

La prima è riservata alla memorizzazione del codice dei caratteri e la seconda alla memorizzazione del codice dei colori.

A questo proposito è consigliabile, prima di andare avanti nella lettura dell'articolo, leggere con attenzione le semplici tabelle delle appendici "E" e "G" del manuale d'istruzione del C-64.

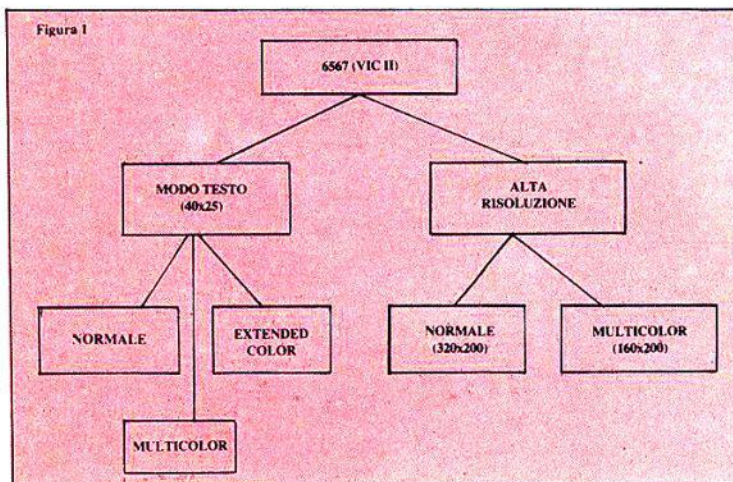
Proviamo a digitare, dopo aver spento e riacceso il computer:

```
POKE 1024,1: POKE 55296,0
```

A questo punto dovrebbe apparire, nell'angolo in alto a sinistra, il carattere "A" di colore nero; infatti 1 è il codice della A e 0 è il codice del colore nero. I caratteri rappresentabili sullo schermo sono in tutto 256, i colori 16.

Provate a far girare questo programmino per rendervene conto:

```
10 FOR I=0 TO 255:
   POKE 1024+I,I: NEXT
20 FOR I=0 TO 15: FOR K=0 TO 255
30 POKE 55296+K,I: NEXT
40 FOR T=1 TO 1000: NEXT T,I
```

Saranno visualizzati, in alto sullo schermo, i 256 caratteri nei 16 colori possibili. L'organizzazione della memoria video, che inizia alla locazione 1024 compresa, è strutturata in modo che ciascun byte rappresenti la posizione a destra del

byte precedente e, ovviamente, quella a sinistra del byte successivo. Gli indirizzi che rappresentano le singole caselle contenenti i caratteri e i colori aumentano (da 1024 a 2023) a mano a mano che da sinistra ci si sposta a destra e dall'alto in basso sullo schermo (consultare le tabelle nell'appendice G del manuale).

Lo schermo è pertanto costituito da 25 righe contenenti ognuna 40 caratteri, per un totale di 1000 singole posizioni che possiamo riempire a piacimento con un diverso colore ed un diverso carattere.

Se abbiamo l'esigenza di scrivere un carattere con un certo colore in una determinata posizione dello schermo, utilizzeremo la seguente subroutine, dopo aver posto le variabili CA (codice carattere), CO (codice colore), RI (numero riga) e CL (numero colonna). Essa può funzionare in qualsiasi programma BASIC e, per utilizzarla, basta predisporre le variabili sopracitate ai valori desiderati e chiamarla con GOSUB 60000.

```
5 PRINT CHR$(147)
10 INPUT"CARATTERE (0-255)";CA
20 INPUT"COLORE (0-15)";CO
30 INPUT"RIGA (0-24)";RI
40 INPUT"COLONNA (0-39)";CL
50 GOSUB 60000: GOTO 5
60000 PRINTCHR$(147):
PO=RI*40+CL
60010 POKE1024+PO,CA:
POKE 55296,PO,CO
60020 GETA$:IFA$=""THEN60020
60030 RETURN
```

I valori da assegnare alle variabili devono essere numeri interi e non oltrepassare, in più o in meno, quelli segnalati tra parentesi onde evitare brutte sorprese.

Come spostare la memoria di schermo

Prima di leggere questo paragrafo spegniamo e riaccendiamo il computer, quindi digitiamo:

```
POKE 44,64: POKE 64*256,0: NEW
```

Ciò servirà a separare la memoria del BASIC da quella del video, onde evitare sgradevoli inconvenienti.

Grazie alla versatilità del chip 6567, si possono ottenere molte opzioni di visualizzazione "fuori-standard" e una di queste permette di spostare la memoria dello schermo in più parti di memoria RAM.

La locazione 53272 rappresenta il registro del VIC II atto a modificare l'indirizzo iniziale della memoria di schermo. Costituisce, insomma, un pro-memoria che il C-64 consulta sempre prima di visualizzare qualsiasi cosa.

Questo registro è diviso in due nibble (mezzo byte o quattro bit, sono detti anche nibble) dei quali interessa quello alto (il nibble alto è quello che comprende i bit di posizione 7, 6, 5, 4, mentre quello basso è costituito dai bit 3, 2, 1, 0.

Variando opportunamente il nibble alto, si può cambiare la zona di memoria dello schermo in accordo con la seguente tabella:

Maschera	Byte 53272	Inizio schermo	Posizione
0	00000101	0	0
16	00010101	1024	1
32	00100101	2048	2
48	00110101	3072	3
64	01000101	4096	4
80	01010101	5120	5
96	01100101	6144	6
112	01110101	7168	7
128	10000101	8192	8
144	10010101	9216	9
160	10100101	10240	10
176	10110101	11264	11
192	11000101	12288	12
208	11010101	13312	13
224	11100101	14336	14
240	11110101	15360	15

Se, ad esempio, vogliamo fare in modo che l'area di memoria di schermo non sia più quella consueta che inizia a 1024, ma tale che abbia inizio a partire da 2048, dobbiamo digitare:

POKE 44,64:POKE 64*256,0:NEW

Subito dopo, digitate il mini-programma:

10 POKE 53272,(PEEK(53272) AND15)OR32:POKE648,8

L'operazione PEEK (53272) AND 15 serve a porre a 0 tutti i bit del nibble alto in modo da "prepararli" all'operazione OR 32 che accende il bit 5 posizionando lo schermo a partire da 2048, come, appunto, rintracciabile in tabella. Si noti, tra l'altro, che il nibble basso vale sempre 0101 nella tabella riportata.

Dando il RUN al mini programma di prima, dovrebbero comparire dei caratteri senza senso sullo schermo e il cursore lampeggerà al di sotto del solito READY (difficile, in verità, da individuare nel caos generale).

L'istruzione POKE 648,8 serve ad "avvertire" il computer che si è cambiato lo schermo; il valore da "POKEare" nella locazione 648 è dato dal rapporto fra l'indirizzo della prima cella video e il valore 256:

Valore da pokare = nuovo inizio/256

Infatti 2048/256 fornisce, appunto, 8.

Ma torniamo al nostro schermo che è diventato un'accozzaglia di caratteri dopo il RUN.

Usate i tasti SHIFT e CLR/HOME per cancellare e, subito dopo, date il LIST. Sembrerebbe che tutto funzioni a dovere.

Ma provate ora ad eseguire:

POKE 1024,1

Ci aspetteremmo il carattere "A" in alto a sinistra, ma non compare.

Proviamo allora a digitare:

POKE 2048,1

Eccola, la "A". Ciò dimostra che col trucchetto di prima abbiamo spostato la memoria di schermo dalla solita area 1024-2023 alla zona RAM 2048 3047.

Provate, ora, a premere RUN/STOP e RESTORE...

E' ricomparsa la schermata di prima, e digitando tasti a casaccio, non compare nulla.

A questo punto eseguite, nell'ordine:

- premete il tasto RETURN (per "uscire" da un eventuale modo virgolette);
- premete SHIFT e CLR/HOME (anche se sembra che non accada nulla di nulla);
- digitate (alla "cieca") RUN e il tasto RETURN...

Miracolo! Siamo tornati come prima e tutto funziona di nuovo normalmente, naturalmente con lo schermo posizionato da 2048 a 3047.

Visualizzazione caratteri nel modo testo

Prima di passare alla lettura di questo paragrafo, spegniamo e riaccendiamo il 64 in modo da riposizionare il video in condizioni "normali".

I caratteri del Commodore 64 sono definiti, ciascuno, da una sequenza di byte divisi in gruppi di otto. Questi sono stati registrati in fabbrica dalla Commodore su di una ROM interna al computer e occupano, in totale, 4 Kbyte (poco più di 4000 byte ed esattamente da 53248 a 57343).

Il carattere che rappresenta la lettera "A", ad esempio, è situato negli otto byte allocati da 53256 a 53263; il valore binario di questi ultimi è il seguente:

Nel modo testo normale, ogniqualvolta il VIC II deve visualizzare la lettera "A", preleva questi otto byte, li legge bit per bit e traccia un puntino (dot) con il colore dello sfondo (sarà cioè invisibile) se il bit appena analizzato vale 0. Il puntino assumerà invece il colore il cui codice è situato nella relativa casella del colore se il bit vale invece 1.

La grafica in alta risoluzione

Nel modo testo normale, il VIC II legge i byte residenti in ROM per trascrivere i caratteri sullo schermo, ma se volessimo fare un disegno oppure, più in generale, accendere o spegnere questo o quel puntino (dot) in una qualsiasi parte dello schermo, dovremmo ricorrere alla grafica in alta risoluzione (HRG = High Resolution Graphic).

Questo modo di rappresentazione, unito al modo testo, conferisce al Commodore 64 quella versatilità d'impiego che nessun computer della stessa fascia può eguagliare. La differenza tra il modo testo ed il modo HRG consiste, principalmente, nella quantità di memoria usata. Questa, nel caso del modo testo, è composta da 1000 byte RAM (le locazioni che compongono lo schermo) e da 4096 byte ROM (sono i byte che definiscono i due set dii256 caratteri). Se, al contrario, si ricorre all'HRG, la memoria necessaria è di 8000 byte RAM. Abbiamo tralasciato di conteggiare i 1000 byte usati dalle celle colore perchè la

Locazione	Valore	Carattere
53256	00011000	**
53257	00111100	****
53258	01100110	** **
53259	01111110	******
53260	01100110	** **
53261	01100110	** **
53262	01100110	** **
53263	00000000	

gestione e la posizione di queste ultime cambia notevolmente da un modo all'altro.

Nel modo HRG ogni puntino corrisponde a un bit di memoria RAM ed è, pertanto, modificabile grazie alla manipolazione del byte cui appartiene.

Lo schermo del C-64 è formato da una griglia di 40x25 caratteri; essendo ogni carattere composto da 8x8 puntini, avremo una risoluzione di 320 (40x8) puntini orizzontalmente e di 200 (25x8) verticalmente, in definitiva 64000 dot facenti parte di 8000 byte (8000x8 =64000).

Indirizzo	Contenuto	Rappresentazione sullo schermo (Angolo in alto a sinistra)	
8192	01011001 →	* ** *	* * * ←
8193	00101100 →	* **	* * * * ←
8194	00010110 →	* **	.... ←
8195	00001011 →	* **	
8196	10110101 →	* ** * *	
8197	01011010 →	* ** *	
8198	00101101 →	* ** *	
8199	00010110 →	* **	
8200	01010010	------------	
8201	10100101	------------	
....	....	------------	

I colori in Hi-Res

I colori disponibili sono due per ogni cella 8x8, il primo da assegnare ai bit che valgono zero e l'altro per i bit che valgono uno.

Nel modo HRG la tabella dei colori non risiede più nei mille byte della zona 55296-56295, ma nell'area di memoria localizzata da 1024 a 2023. Ciò significa che le informazioni relative al colore vengono a "sovrapporsi" alla stessa parte di memoria ove risiedeva lo schermo nel modo testo. Se, in altre parole, sullo schermo abitualmente usato per il testo scriviamo una frase qualunque, ciascun carattere di questa frase assume il significato di un codice per colorare la cella video quando questo è visualizzato nel modo alta risoluzione.

Il carattere "A" pokato, ad esempio, nella prima cella video in alto a sinistra, ha il codice 1 (provate a digitare POKE 1024,1). Poichè tale valore "1" corrisponde al numero binario 00000001 oppure, se preferite, ai due nibble 0000 e 0001, si avrà, una strana conseguenza.

I 64 dot di alta risoluzione, corrispondenti agli otto byte dell'angolo in alto a sinistra, assumeranno il colore del nibble basso (bianco, nel caso particolare del carattere "A") oppure quello del nibble alto (nero, sempre nel caso della lettera "A") a seconda se i singoli bit, del gruppo di otto byte in Hi-Res, sono settati oppure resettati.

Naturalmente, dato che un nibble è composto da quattro bit e che con questi è possibile rappresentare un numero variabile da zero (0000) a 15 (1111), il codice dei 16 colori disponibili è proprio quello dei 16 colori Commodore (0=nero, 1=bianco, eccetera).

Ad esempio, il carattere "P", ottenuto con POKE1024,16 è costituito dai nibble 0001 e 0000. Gli otto byte che dipendono (in Hi-Res) da tale carattere - codice avranno, come colori, il bianco o il nero a seconda se settati o resettati.

Il fatto, dunque, che in HRG l'area colore coincida con quella abituale del testo, costituisce una seccatura che, in qualche modo, deve essere superata.

La struttura della memoria di schermo viene quindi totalmente rivoluzionata: il circuito video VIC II non interpreta più il singolo byte, situato sullo schermo (o meglio, zona 1024 - 2023), come uno dei 256 codici dei caratteri il cui disegno è residente in ROM. Si limita a considerarlo come un insieme di due nibble-codice da cui trarre informazioni per i colori da assegnare agli 8x8 puntini delle mini zona in Hi-Res.

Per ognuno dei 64000 bit dell'alta risoluzione, il calcolatore assegna il colore definito dal nibble alto della relativa cella di colore (ricordiamo che ora le celle del colore risiedono da 1024 a 2023) se questo vale 1 e quello definito dal nibble basso vale 0.

Lo schermo in HRG deve essere considerato come l'immagine fisica della memoria RAM nella quale è situato lo schermo, per cui, modificando opportunamente i byte implicati, si ottengono disegni e grafici nettamente migliori di quelli che si possono definire con i caratteri grafici nel modo testo.

Gestione dei 64000 punti

La memoria dello schermo occupa 8000 byte (vedremo dopo quali possono essere) e si divide in 200 righe per 320 colonne. L'integrato 6567 legge, di queste 8000 locazioni, un byte per volta ricavandone 8 bit e trasformandoli in 8 puntini. In seguito legge il successivo e pone i suoi bit al di sotto di quelli del byte precedente.

Il nono byte, invece, lo pone a destra del primo, il decimo a destra del secondo, l'undicesimo a destra del terzo e così via, fino a completare la prima striscia orizzontale che, alla fine, ha le dimensioni di 320 puntini orizzontali e 8 verticali.

A questo punto va "a capo" a sinistra, immediatamente sotto l'ottavo byte, ripetendo ciò che è avvenuto per il primo ciclo, finchè non riempie tutto lo schermo.

Ammettendo che la memoria video incominci da 8192, la sequenza seguita dal VIC II nel leggere i byte è mostrata da questa figura:

8192	8200	8208
8193	8201	8209
8194	8202	8210
8195	8203	8211
8196	8204	8212
8197	8205	8213
8198	8206	8214
8199	8207	8215

Le prime tre celle video e la rispettiva posizione dei 24 byte corrispondenti

Per accedere al modo RHG bisogna porre a 1 il bit 5 della locazione 53265 con il comando:

POKE 53265, PEEK (53265) OR 32

Per disattivarlo, basta digitare:

POKE 53265, PEEK (53265) AND 223

E' doveroso ricordare che, attivando il modo HRG subito dopo l'accensione o, comunque, in condizioni "normali" di funzionamento, la pagina dei colori risiederà da 1024 a 2023 (come già detto), mentre quella dello schermo sarà automaticamente allocata, purtroppo, da 0 a 7999.

Le locazioni che vanno da 0 a 1023, infatti riservate al BASIC e al sistema operativo. Pertanto, per evitare inevitabili quanto improvvisi blocchi o malfunzionamenti, è assolutamente indispensabile spostare l'area RAM dello schermo HRG in un'altra zona di memoria, in modo da non entrare in conflitto col BASIC.

Lo schermo dell'HRG può essere spostato, con molta semplicità, nella zona di memoria che si estende da 8192 a 16191 mettendo a 1 il bit 3 della locazione 53272:

POKE 53272, PEEK (53272) OR 8

Nel caso si volesse utilizzare, invece, la zona che va da 0 a 7999, bisognerà digitare:

POKE 53272, PEEK (53272) OR 1

ma non fatelo mai...

I programmi

A conclusione sono state approntate delle routine (riunite tra loro in un solo programma DEMO) che possono essere inserite in qualsiasi programma BASIC, oltre a un giochino e alla classica rappresentazione grafica della funzione sinusoidale.

Queste routine (o, meglio, subroutine), per le loro caratteristiche, devono risiedere, insieme al programma principale che le accoglie, in una zona di memoria al di sopra dell'indirizzo 16383.

Ciò corrisponde a spostare l'area RAM destinata ad accogliere i programmi BASIC.

Pertanto, prima di digitarle o leggerle da nastro o da disco, occorre eseguire:

POKE44,64:POKE64*256,0:NEW

Bisogna stare molto attenti a questo, perchè si rischia di perdere ore di lavoro solo per non aver eseguito la linea di cui sopra.

Le specifiche delle singole subroutine sono le seguenti:

60000: questa routine serve per attivare la pagina grafica in alta risoluzione e ha bisogno di due parametri: il codice colore dei puntini accesi (variabile CC) e il codice colore di quelli spenti, ossia lo sfondo (variabile CS).

Per usarla occorre assegnare un valore compreso tra 0 e 15 sia alla variabile CC che a CS. Subito dopo si esegue semplicemente GOSUB 60000;

61000: questa routine serve per disattivare la pagina grafica in alta risoluzione e non ha bisogno di parametri.

Per usarla occorre un semplice: GOSUB 61000;

62000: questa routine serve per tracciare un singolo puntino (dot) alle coordinate definite dalle variabili X ed Y. Nel porre queste variabili, è bene sapere che il primo punto in alto a sinistra sullo schermo ha come coordinate X=0 ed Y=0 e quello in basso a destra X=319 ed Y=199.

Per usarla occorre assegnare un valore compreso tra 0 e 319 alla variabile X e uno compreso tra 0 e 199 a Y. In seguito: GOSUB 62000;

62500: questa routine serve per tracciare una linea dal punto di coordinate X1, Y1 al punto di coordinate X2, Y2. Notate come in questo listato sia utilizzata anche la routine 62000: è infatti quest'ultima che traccia fisicamente la linea.

Per usarla, X1 e X2 devono esser comprese tra 0 e 319, Y1 e Y2 tra 0 e 199. In seguito basta un: GOSUB 62500;

63000: questa routine serve per riempire (e, anche, cancellare) lo schermo grafico: non è rapidissima (in quanto deve pokare ben 8000 byte) ma decisamente "didattica".

E' consigliabile, onde evitare sgradevoli effetti grafici, chiamarla prima di utilizzare la routine che attiva l'HRG.

Per usarla: GOSUB 63000.

Nell'utilizzare le routine nei propri programmi è bene assegnare alle variabili valori interi e nei limiti indicati predentemente.

Un esempio di applicazione delle routine è dato dal giochino "Equilibrista" che le utilizza per disegnare lo scenario in cui si svolge l'azione.

L'ultimo listato disegna una sinusoide, funzione obbligatoria per chi si avvicina per la prima volta al mondo dell'Hi-Res.

Prima di digitarli, o di caricarli, non dimenticare di digitare la linea:

POKE 44,64: POKE 16384,0: NEW

Conclusione

Ciò che abbiamo spiegato serve a porre il lettore in grado di capire, almeno in parte, i meccanismi interni che regolano la visualizzazione dei dati nel Commodore 64.

Forse il discorso è incompleto, ma abbiamo optato per una ricostruzione essenziale dei vari aspetti inerenti al problema della grafica, dedicando più spazio a quello che di questa branca, si rivela essere l'argomento più interessante.

Seguendo questo modo di procedere, non è escluso che nei prossimi numeri possiate trovare articoli dedicati a temi specifici, come la gestione degli sprites e dei caratteri programmabili, oppure approfondimenti di argomenti toccati solo in superficie.

Bit, byte e sistema binario

La memoria del C-64 è composta da 65536 byte (celle, locazioni di memoria) atti a contenere tutti i dati e i programmi necessari a farlo funzionare. Il byte è, in parole molto povere, l'unità di memoria del calcolatore ed è formato, a sua volta, da 8 bit.

Il sistema di numerazione più adatto a rappresentare il valore di un byte è quello binario, proprio perchè utilizza due soli simboli numerici, cioè le cifre "0" e "1" (la parola "bit" è l'abbreviazione di "BInary digiT", ossia "cifra binaria").

Ecco l'esempio di un byte scomposto nei suoi 8 bit:

	Byte
Bits	01101001
Posizione	76543210

Il sistema binario, come quello decimale, adotta una notazione di tipo "posizionale" tale, cioè, che attribuisce una diversa "importanza" a seconda della posizione che una cifra occupa all'interno di un numero.

Si può "tradurre" in decimale qualsiasi numero binario sommando i valori ottenuti moltiplicando ogni cifra per la base elevata alla posizione occupata dalla cifra stessa.

In altre parole, per trasportare il byte, portato come esempio, in decimale, dovremo effettuare le seguenti operazioni:

Cifra		Base		Posiz.				
0	*	2	↑	7	=0	*	128 =	0 +
1	*	2	↑	6	=1	*	64 =	64 +
1	*	2	↑	5	=1	*	32 =	32 +
0	*	2	↑	4	=0	*	16 =	0 +
1	*	2	↑	3	=1	*	8 =	8 +
0	*	2	↑	2	=0	*	4 =	0 +
0	*	2	↑	1	=0	*	2 =	0 +
1	*	2	↑	0	=1	*	1 =	1 =
								105

Gli operatori logici

Quando immettiamo, con l'istruzione POKE, un valore in una delle celle, il dato che vi era contenuto viene cancellato e sostituito dal nuovo valore, ma a volte si desidera cambiare solo alcuni bit che lo formano. In questo caso ci vengono incontro le istruzioni "AND" e "OR", dette anche operatori logici.

Entrambe, come l'addizione ordinaria, hanno bisogno di due operandi su cui agire e danno un risultato che nulla ha a che vedere con le normali operazioni aritmetiche.

Uno degli operandi è chiamato anche "maschera" e serve per definire quale bit vogliamo modificare e quale vogliamo lasciare invariato. Il lettore noti come, nel gergo informatico, il bit può essere posto a "1" (settato) oppure a "0" (resettato), o anche acceso o spento, alto o basso...

L'altro operando non è altro che il valore primitivo del byte da cambiare.

L'operatore AND confronta uno ad uno i bit dei due operandi che occupano la stessa posizione formulando il risultato in base alla seguente tabella:

Tabella dell'And

	0	1
0	0	0
1	0	1

Dalla tabella possiamo estrapolare la funzione AND: esso è adatto, avendo creato una maschera opportuna, a porre a zero determinati bit di un byte.

Se, ad esempio, vogliamo spegnere i soli bit di posizione "0" e "4" di un qualsiasi byte (lasciando inalterati gli altri), dobbiamo creare una maschera di 8 cifre (bit) che abbia solo questi bit a 0, leggere il byte dalla memoria, fare l'AND di quest'ultimo con la maschera e riporlo di nuovo in memoria. In termini pratici, la maschera sarà:

Bit	11101110
Posizione	76543210

il cui corrispondente decimale, calcolato con il metodo descritto nel precedente paragrafo, sarà 238 che deriva dalle seguenti somme:

2 elevato a 7:	128
2 elevato a 6:	64
2 elevato a 5:	32
2 elevato a 3:	8
2 elevato a 2:	4
2 elevato a 1:	2
	238

A questo punto, ponendo che il byte da modificare risieda alla locazione 2024, si esegue:

```
POKE 2024, PEEK (2024) AND 238
```

Se il valore primitivo del byte preso in considerazione fosse stato 01100011, facendo l'AND con la maschera 11101110, il valore ottenuto sarebbe stato 01100010 (98). Infatti:

N.	Byte da mascherare		Masch.		Risul.	
7	0	AND	1	=	0	
6	1	AND	1	=	1	
5	1	AND	1	=	1	
4	1	AND	0	=	0	← bit modif.
3	0	AND	1	=	0	
2	0	AND	1	=	0	
1	1	AND	1	=	1	
0	1	AND	0	=	0	← bit modif.

Come possiamo notare, il risultato è uguale al valore primitivo dei byte tranne che per i bit 4 e 0, che sono stati posti a 0.

L'operatore OR è il complementare di AND, in quanto permette di accendere singoli bit di un byte. La tabella mediante la quale i bit sono trasformati da OR è la seguente:

⟶

Tabella OR

	0	1
0	0	1
1	1	1

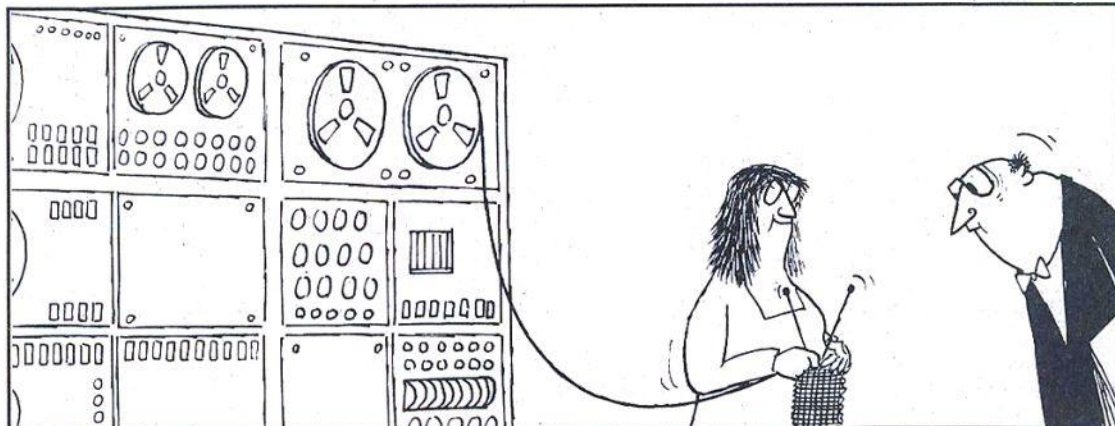
Anche in questo caso, bisogna creare una maschera che, al contrario di quella per l'AND, deve avere tutti i bit a 0 eccetto quelli che vogliamo modificare. La maschera da utilizzare nel caso volessimo porre a 1 il bit 7 è:

Bit	11101110
Posizione	76543210

che equivale a 128 decimale. Prendendo come esempio il byte utilizzato in precedenza, si avrà il seguente risultato:

N.	Byte Da mascherare		Maschera		Risultato	
7	0	OR	1	=	1	← bit modificato
6	1	OR	0	=	1	
5	1	OR	0	=	1	
4	1	OR	0	=	1	
3	0	OR	0	=	0	
2	0	OR	0	=	0	
1	1	OR	0	=	1	
0	1	OR	0	=	1	

Tutto ciò si traduce in Basic con:

```
POKE 2024, PEEK (2024) OR 128
```

```
  90 PRINT"[CLEAR]TI SEI RICORDA
     TO DI DIGITARE:"
  92 PRINT:PRINT"POKE 44,64:POKE
      64*256,0:NEW"
  93 PRINT:PRINT"(PREMI: S/N)"
  94 GET A$:IF A$="" THEN 94
  95 IF A$<>"S" THEN END
 100 PRINTCHR$(147)"QUANDO IL CO
     LORE DEL BORDO CAMBIERA'":P
     OKE 53280,0
 110 PRINT"VORRA' DIRE CHE LA DI
     MOSTRAZIONE E' TERMINATA"
 120 PRINT"A QUESTO PUNTO PREMER
     AI UN TASTO"
 130 PRINT"PER TORNARE AL MENU P
     RINCIPALE":PRINT:PRINT
 131 PRINT"1- RIEMPIE SCHERMO HI
     -RES"
 132 PRINT"2- ASSEGNA COLORE AI
     NIBBLE"
 133 PRINT"3- TRACCIA LINEA IN H
     I-RES"
 134 PRINT"4- TRACCIA STRISCIA I
     N HI-RES"
 135 GET A$:IF A$="" THEN 135
 136 IF A$="2" THEN GOSUB 1000:G
     OTO 100
 140 IF A$="1" THEN GOSUB 2000:G
     OTO 100
 150 IF A$="3" THEN GOSUB 3000:G
     OTO 100
 160 IF A$="4" THEN GOSUB 4000:G
     OTO 100
 170 GOTO 100
1000 PRINT:INPUT "COL.PUNTI SETT
     ATI (0-15)";CS:IF CS<0 OR C
     S>15 THEN RETURN
1010 PRINT:INPUT "COL.PUNTI RESE
     TTATI (0-15)";CC:IF CC<0 OR
      CC>15 THEN 1000
1015 IF CC=CS THEN PRINT"COLORI
     EGUALI!":GOTO 1000
1020 PRINTCHR$(147):GOSUB 60000:
     GOSUB 60500:POKE 53280,1
1030 GET A$:IF A$="" THEN 1030
1040 GOSUB 61000:RETURN
2000 INPUT "COLORE SET";CC:IF CC
     <0 OR CC>15 THEN RETURN
2010 INPUT "COLORE RESET";CS:IF
     CS<0 OR CS>15 THEN 2000
2015 IF CS=CC THEN PRINT"COLORI
     EGUALI!":GOTO 2000
2020 PRINT"CARATTERE (0=CANCELLA
      255 RIEMPIE"
2025 INPUT CR:IF CR<0 OR CR>255
     THEN 2010
2030 GOSUB 60000:GOSUB 60500:GOS
     UB 63000:POKE 53280,1
2040 GET A$:IF A$="" THEN 2040
2050 GOSUB 61000:RETURN
3000 IF CS=CC THEN PRINT"HAI DIM
     ENTICATO DI SETTARE I COLOR
     I!":GOTO 3050
3005 INPUT "X1= (0-319)";X1:IF X
     1<0 OR X1>319 THEN 3000
3010 INPUT "Y1= (0-199)";Y1:IF Y
     1<0 OR Y1>199 THEN 3000
3020 INPUT "X2= (0-319)";X2:IF X
     2<0 OR X2>319 THEN 3020
3030 INPUT "Y2= (0-199)";Y2:IF Y
     2<0 OR Y2>199 THEN 3030
3040 POKE 53280,0:GOSUB 60000:GO
     SUB 60500:GOSUB 62500:POKE
     53280,1
3050 GET A$:IF A$="" THEN 3050
3060 GOSUB 61000:RETURN
4000 INPUT "COLORE SET";CS:IF CS
     <0 OR CS>15 THEN RETURN
4002 INPUT "COLORE RESET";CC:IF
     CC=CS THEN PRINT"COLORI EGU
     ALI!":GOTO 4000
4003 IF CC<0 OR CC>15 THEN 4002
4005 INPUT "QUALE STRISCIA (0-24
     )";NL:IF NL<0 OR NL>24 THEN
      4002
4007 INPUT "CHE CARATTERE (0-255
     )";CR:IF CR<0 OR CR>255 THE
     N 4005
4010 INPUT "VELOCITA' (0-100)(0=
     MAX)";VE:IF VE<0 OR VE>100
     THEN 4007
4015 GOSUB 60000:GOSUB 60500:FOR
      I=1024+NL*40 TO 1024+40*NL
     +39
4020 POKE I,CC*16+CS:NEXT
4030 FOR I=8192+NL*8*40 TO 8191+
     8*40*NL+320:POKE I,CR
4040 FOR J=1 TO VE:NEXTJ,I:POKE
     53280,1
4050 GET A$:IF A$="" THEN 4050
```

```
 4060 GOSUB 61000:RETURN
59999 END
60000 POKE 53272,PEEK(53272) OR 8
60010 POKE 53265,PEEK(53265) OR 3
      2:RETURN
60500 CT=CC*16+CS:FOR I=1024 TO 2
      023:POKE I,CT:NEXT:RETURN
61000 POKE 53272,21:POKE  53265,2
      7:RETURN
62000 IF (X<0) OR (X>319) OR (Y<0
      ) OR (Y>199) THEN RETURN
62010 BY=8192+INT(Y/8)*320+8*INT(
      X/8)+(Y AND 7)
62020 POKE BY,PEEK(BY) OR (2↑(7-(
      X AND 7))):RETURN
62500 XD=X2-X1:YD=Y2-Y1
62510 A0=1:A1=1
62520 IF YD<0 THEN A0=-1
62530 IF XD<0 THEN A1=-1
62540 XE=ABS(XD):YE=ABS(YD)
62550 IF (XE-YE)>=0 THEN 62590
62560 S0=-1:S1=0:LG=YE:SH=XE
62570 IF YD>=0 THEN S0=1
62580 GOTO 62610
62590 S0=0:S1=-1:LG=XE:SH=YE
62600 IF XD>=0 THEN S1=1
62610 UD=LG-SH:X=X1:Y=Y1
62620 FOR TH=LG TO 0 STEP -1:GOSU
      B 62000
62630 IF TC>=0 THEN 62660
62640 TC=TC+SH:X=X+S1:Y=Y+S0
62650 GOTO 62670
62660 TC=TC-UD:X=X+A1:Y=Y+A0
62670 NEXT:RETURN
63000 FOR I=8192 TO 16191:POKE I,
      CR:NEXT:RETURN
```

```
   10 REM  DISEGNO DI UNA SINUSOI
      DE
   20 REM  IN ALTA RISOLUZIONE
   40 REM  A T T E N Z I O N E
   60 REM  DIGITARE:
   70 REM  POKE44,64:POKE16384,0:
      NEW
   80 REM  PRIMA DI LEGGERE
   85 REM  O BATTERE IL PROGRAMMA
   90 :
   91 REM  SOFTWARE DI
   92 REM  PASQUALE  D'ANDRETI
   99 :
  100 IF PEEK(44)=64 THEN 200
  110 PRINTCHR$(147)"DIGITA:":PRI
      NT
  120 PRINT"POKE 44,64: POKE16384
      ,0: NEW":PRINT:PRINT
  130 PRINT"E RILEGGI O RIBATTI I
      L PROGRAMMA":END
  200 PRINTCHR$(147):PRINT:PRINT"
      ATTENDI 16191"
  210 C=100:P=90:V=20:GOSUB 63000
  220 CS=15:CC=0:GOSUB 60000
  230 FOR X=0 TO 319 STEP 1
  240 Y=INT(C+P*SIN(X/V))
  250 GOSUB 62000:NEXT
  260 POKE 198,0:WAIT 198,1
  270 GOSUB 61000:END
59990 REM  ***
59991 REM  *** ATTIVA GRAFICA E C
      OLORI
59992 REM  ***
60000 POKE 53272,PEEK(53272) OR 8
60010 POKE 53265,PEEK(53265) OR 3
      2
60020 CT=CC*16+CS:FOR I=1024 TO 2
      023:POKE I,CT:NEXT
60030 RETURN
60990 REM  ***
60991 REM  *** DISATTIVA GRAFICA
60992 REM  ***
61000 POKE 53265,27:POKE 53272,21
61010 PRINTCHR$(147):RETURN
61990 REM  ***
61991 REM  *** DISEGNA PUNTO (X,Y
      )
61992 REM  ***
62000 IF (X<0) OR (X>319) OR (Y<0
      ) OR (Y>199) THEN RETURN
62010 BY=8192+INT(Y/8)*320+8*INT(
      X/8)+(Y AND 7)
62020 POKE BY,PEEK(BY) OR (2↑(7-(
      X AND 7))):RETURN
62990 REM  ***
62991 REM  *** CANCELLA SCHERMO G
      RAFICO
62992 REM  ***
63000 FOR I=8192 TO 16191:POKE I,
      0:PRINTCHR$(19)I:NEXT
63010 RETURN
```

```
 20 REM  L' EQUILIBRISTA PER C-
    64
 40 REM  A T T E N Z I O N E
 50 REM  DIGITARE: POKE 44,64:
    POKE 16384,0: NEW
 52 REM  PRIMA DI BATTERE
 55 REM  O LEGGERE IL PROGRAMMA
 65 REM  SOFTWARE DI
 70 REM  PASQUALE D' ANDRETI
 75 REM  RIARDO (CE)

        *
 90 :
100 IF  PEEK (44) = 64  THEN  1
    50
110 PRINTCHR$(147)"DIGITA:":PRI
    NT:PRINT
120 PRINT "POKE 44,64: POKE1638
    4,0: NEW"
130 PRINT:PRINT:PRINT"E RILEGGI
     O RIBATTI IL PROGRAMMA": E
    ND
150 GOSUB  50000: POKE  650,128
200 POKE  53280,0: POKE  53281,
    0: GOSUB  57000
210 FOR  I = 0  TO  127: READ
    S:POKE 832 + I, S: NEXT
220 GOSUB  63000: CC = 0: CS =
    15: GOSUB  60000
230 Y = 25: FOR  X = 30  TO  28
    9: GOSUB  62000: NEXT
240 X = 30: FOR  Y = 25  TO  10
    0: GOSUB  62000: NEXT
250 X = 289: FOR  Y = 25  TO  1
    00: GOSUB  62000 : NEXT
260 X1 = 0: Y1 = 100: X2 = 30:
    Y2 = 25: GOSUB  62500
270 X1 = 319: Y1 = 100: X2 = 28
    9: Y2 = 25: GOSUB  62500
280 X1 = 0: Y1 = 62: X2 = 30: Y
    2 = 25: GOSUB  62500
290 X1 = 289: Y1 = 25: X2 = 319
    : Y2 = 62: GOSUB  62500
300 X1 = 0: Y1 = 0: X2 = 30: Y2
     = 25: GOSUB  62500
310 X1 = 319: Y1 = 0: X2 = 289:
     Y2 = 25: GOSUB  62500
320 GOSUB  58000
400 POKE  2040,13: POKE  2041,1
    4: POKE  2042,14: POKE  204
    3,14
410 POKE  53287,1: POKE  53288,
    11: POKE  53289,12: POKE  5
    3290,2
420 POKE  53277,0: POKE  53271,
    0
430 POKE  53249,54
440 POKE  53250,90
441 POKE  53252,172
442 POKE  53254,255
450 POKE  53269,15
500 S1 = 53248: S2 = S1 + 3: S3
     = S2 + 2: S4 = S3 + 2: PM
    = 87
502 P1 = 250: P2 = P1: P3 = P1:
     Z = P1: W = 225: S = 7
505 O = 8: PD = 249: PS = 93: F
    C = 53278: PU=0
507 FOR  I = 1  TO  700
510 GET A$:IF A$="L" AND (PM<PD
    ) THEN PM=PM+S
520 IF  A$ = "J"  AND  (PM > PS
    )  THEN  PM = PM - S
530 IF  P1 = Z  AND  (RND (.) >
     .9)  THEN  P1 = W
540 IF  P2 = Z  AND  (RND (.) >
     .7)  THEN  P2 = W
550 IF  P3 = Z  AND  (RND (.) >
     .8)  THEN  P3 = W
560 P1 = P1 + ( (P1 < Z) * S)
570 P2 = P2 + ( (P2 < Z) * S)
580 P3 = P3 + ( (P3 < Z) * S)
590 POKE  S1,PM: POKE  S2,P1: P
    OKE  S3,P2: POKE  S4,P3
600 P1 = - (P1 < O) * Z - (P1 >
     S) * P1
601 P2 = - (P2 < O) * Z - (P2 >
     S) * P2
602 P3 = - (P3 < O) * Z - (P3 >
     S) * P3
610 IF  PEEK (FC) <> 0  THEN  P
    U = PU + 1
620 NEXT
630 GOSUB  61000: POKE  53269,0
640 PRINTCHR$(147)"HAI FATTO "
    PU "  PUNTI"
650 IF  RO < PU  THEN  670
660 PRINT "IL RECORD DI OGGI E'
     DI " RO "  PUNTI": GOTO  6
    90
```

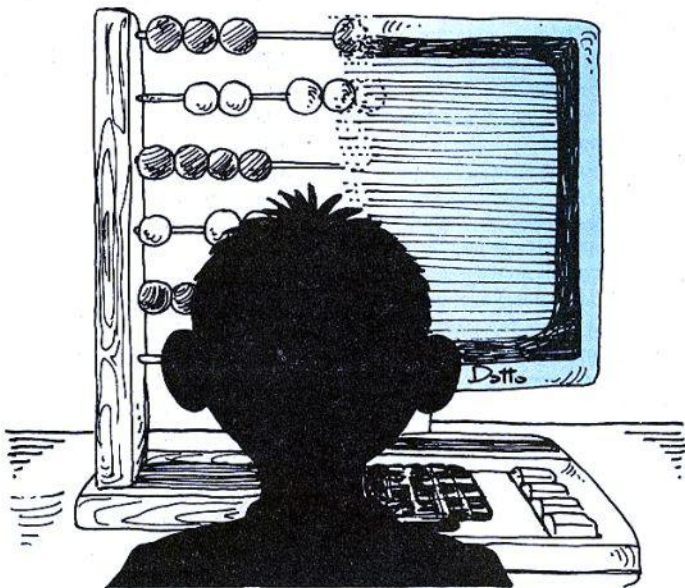
```
 670 PRINT "BRAVO, HAI STABILITO
     UN NUOVO RECORD ! "
 680 PRINT "RECORD: " PU "  PUNT
     I": RO = PU
 690 PRINT "VUOI GIOCARE ANCORA
     (S/N) ?"
 700 FOR  I = 1  TO  4000: NEXT
 710 POKE  198,0: WAIT  198,1: G
     ET  A$
 720 IF  A$ = "S"  THEN  GOSUB
     60000: GOTO  400
 730 END
1000 DATA  0,28,0,0,42,0,0,62,0,
     0,20,0,0,28,0,0
1010 DATA  8,0,0,62,0,0,127,0,0,
     255,128,1,255,192,3,190
1020 DATA  224,7,62,112,2,62,32,
     0,62,0,0,54,0,0,119,0
1030 DATA  0,99,0,0,99,0,0,99,0,
     1,227,192,1,227,192,0
1040 DATA  0,0,0,0,12,0,0,62,0,0
     ,127,0,0,111,0,0
1050 DATA  223,128,0,223,128,0,1
     11,128,0,127,0,0,127,0,0,62
1060 DATA  0,0,28,0,0,8,0,0,8,0,
     0,4,0,0,4,0
1070 DATA  0,8,0,0,8,0,0,8,0,0,8
     ,0,0,4,0,0
50000 B = 8192: O = 8: E = 7: M =
      319: N = 199: T = 320: RET
     URN
57000 PRINTCHR$(147)"
     L' EQUILIBRISTA"
57010 PRINT "FAI MUOVERE L' EQUIL
     IBRISTA SULLA  CORDA"
57020 PRINT "PER FARGLI TOCCARE P
     IU' PALLONCINI  POS-"
57030 PRINT "SIBILE."
57050 PRINT "PREMI IL TASTO J PER
      ANDARE A SINISTRA";
57080 PRINT "ED IL TASTO L PER AN
     DARE A DESTRA."
57100 PRINT "ATTENDI IL CARICAMEN
     TO DEI DATI..."
57110 RETURN
58000 FOR  I = 176  TO  199  STEP
       4: BA = 8192 + INT (I / 8
     ) * 320 + (I  AND  7)
58010 FOR  K = BA  TO  BA + 312
     STEP  8
58020 POKE  K,153: POKE  K + 1,10
     2: POKE  K + 2,102: POKE  K
      + 3,153
58030 NEXT: NEXT: RETURN
60000 POKE  53272, PEEK (53272)
     OR  8
60010 POKE  53265, PEEK (53265)
     OR  32
60020 CT = CC * 16 + CS: FOR  I =
      1024  TO  2023: POKE  I, C
     T: NEXT
60030 RETURN
61000 POKE  53265,27: POKE  53272
     ,21
61010 PRINT "[CLEAR]": RETURN
62000 IF  (X<0)  OR  (X>M)  OR  (
     Y<0)  OR  (Y>N)  THEN  RETU
     RN
62010 BY = B + INT (Y / O) * T +
     O * INT (X / O) + (Y  AND
     E)
62020 POKE  BY, PEEK (BY)  OR  (2
     ↑ (E - (X  AND  E) ) ): RET
     URN
62500 XD = X2 - X1: YD = Y2 - Y1
62510 A0 = 1: A1 = 1
62520 IF  YD < 0  THEN  A0 = -1
62530 IF  XD < 0  THEN  A1 = -1
62540 XE = ABS (XD): YE = ABS (YD
     )
62550 IF  (XE - YE) >= 0  THEN  6
     2590
62560 S0 = -1: S1 = 0: LG = YE: S
     H = XE
62570 IF  YD >= 0  THEN  S0 = 1
62580 GOTO  62610
62590 S0 = 0: S1 = -1: LG = XE: S
     H = YE
62600 IF  XD >= 0  THEN  S1 = 1
62610 UD = LG - SH: X = X1: Y = Y
     1
62620 FOR  TH = LG  TO  0  STEP -
     1:GOSUB 62000
62630 IF TC>=0 THEN 62660
62640 TC=TC+SH:X=X+S1:Y=Y+S0
62650 GOTO 62670
62660 TC=TC-UD:X=X+A1:Y=Y+A0
62670 NEXT:RETURN
63000 FOR  I = 8192  TO  16191: P
     OKE  I,0: NEXT
63010 RETURN
```

VIC 20, C-16, PLUS4, COMMODORE 64, COMMODORE 128

"SPAGHETTI BASIC"

Il tuo primo programma

di Alessandro de Simone

Termina la chiacchierata introduttiva all'uso del computer. Dovrete trarre voi, ovviamente, le conclusioni sulla semplicità d'uso della nuova macchina domestica...

Il mese scorso abbiamo visto una descrizione delle azioni che ognuno di noi compie quando decide di scendere di casa per acquistare un giornale.

Abbiamo anche detto che un programma non è altro che un elenco di azioni da svolgere in successione. Per distinguere la precedente dalla successiva, si ricorre alla loro numerazione crescente.

Anche il calcolatore va programmato allo stesso modo ma, negli esempi che seguiranno, tale numerazione potrà risultare un po' bizzarra. Infatti i numeri che individuano le azioni (o gruppi di azioni da compiere), non saranno consecutivi, nè si partirà dal numero 1. Lo schema visto nella scorsa puntata, per esempio, andrà scritto come segue:

100/ Scendiamo in strada.
110/ Ci rechiamo all'edicola.
245/ Acquistiamo il giornale.
280/ Torniamo a casa.

Il perchè di tale insolita numerazione risiede nel fatto che ci riserviamo la possibilità di inserire, in seguito, alcune fasi che potremmo aver dimenticato. Se riteniamo, ad esempio, che è indispensabile controllare anzitutto il possesso del denaro, potremo agevolmente inserire una riga di azioni numerata con un numero inferiore a 100 (prima riga del caso appena visto). Questa può essere la seguente:

90/ Controllo del possesso del denaro necessario all'acquisto.

Se fossimo partiti direttamente col numero uno, tale inserimento non sarebbe stato possibile se non cambiando la numerazione a tutte le fasi, con notevole perdita di tempo. Analogamente altre azioni possono essere inserite tra altre linee, purchè la numerazione lo consenta.

In ogni caso, anche ricorrendo all'insolito elenco, viene sempre privilegiato l'ordine "logico" nell'eseguire, l'uno dopo l'altro, i vari compiti. L'importante è,

infatti, fare dapprima l'azione che possiede il numero più basso, subito dopo quella col numero immediatamente maggiore e, di seguito, tutte le altre fino all'ultima.

La programmazione con il computer

Come farà il nostro personal computer a sapere se desideriamo eseguire immediatamente alcune operazioni (Command Mode, vedi puntata precedente) oppure se, al contrario, desideriamo impostare un programma?

Semplice: dal numero che digiteremo **prima** di impartire i vari comandi o istruzioni.

Se, infatti, rileggete con attenzione le precedenti puntate, vi accorgerete che nessuna istruzione studiata inizia con un numero. Il fatto stesso che un comando inizi con una lettera dell'alfabeto, comunica al computer che desideriamo venga eseguito non appena si preme il tasto Return che abbreviamo in queste pagine, come di consueto, col carattere maiuscolo R racchiuso tra parentesi (R).

Esempi:

```
A=32                    (R)
PRINT A*B: C=S(3)+4     (R)
```

Il computer, in definitiva, non appena premiamo il tasto di ritorno carrello (R) controlla, come prima cosa, se il cursore si trova all'estremità di una linea che inizia con un carattere alfabetico (qualunque) oppure numerico.

Nel primo caso esegue immediatamente il comando, a meno che, ovviamente, non riconosca un errore di sintassi o di altro genere.

Nel secondo caso il calcolatore **non** esegue i comandi contenuti nella linea numerata nè controlla che la sintassi sia corretta. Egli, pardon, esso, provvede semplicemente a memorizzare la linea-programma appena digitata.

Passiamo a un esempio pratico ricordando di spegnere e riaccendere il computer. Digitate:

```
100 A=250: B=2          (R)
110 PRINT A;B;A*B       (R)
```

Questo è il vostro primo programma!!

La prima riga, numerata con 100, definisce due variabili: "A", cui assegna il valore 250, e "B" che vale 2.

La successiva, numerata con 110, contiene un comando di stampa (PRINT) relativo alle due variabili e al loro prodotto, separate tra loro dal carattere di punto e virgola (;).

Esaminiamo il "ragionamento" seguito dal computer quando abbiamo digitato le due linee: non appena premiamo Return (riga 100), il computer riconosce che il primo carattere della riga è numerico e, di conseguenza, memorizza da qualche parte l'intera linea. La stessa cosa succede quando battiamo la riga 110. Ma... dov'è il risultato dell'elaborazione?

Semplicemente, non c'è! Digitando un programma, infatti, non facciamo altro che battere un pro-memoria di azioni da compiere in seguito.

Una cosa analoga succede quando programmiamo (è il caso di dirlo) le nostre vacanze: se decidiamo, in giugno, di partire il 4 luglio e di fare una tappa a metà strada prima di raggiungere il luogo di villeggiatura, non significa che... siamo già in viaggio. Semplicemente ci organizziamo in modo che, nel giorno stabilito, vengano svolte una serie di azioni senza imprevisti.

La stessa cosa avviene con un computer: le linee-programma battute sono semplicemente un pro-memoria da utilizzare al momento opportuno.

Dove vengono memorizzate le linee?

Provate a cancellare lo schermo del computer mediante la pressione contemporanea dei tasti Shift e Clr/Home. Subito dopo battete:

```
LIST (R)
```

Le due linee appaiono nuovamente sul video. LIST è infatti il comando che consente di visualizzare tutte le linee precedentemente digitate. Ciò dimostra che, sebbene venga cancellato lo schermo, il programma è sempre presente in memoria.

Il comando che esegue le istruzioni del programma è RUN, che in inglese significa: corri, esegui.

Digitando "RUN" e facendolo seguire dal solito tasto di ritorno carrello, il programma va in esecuzione o, come si suol dire, vien fatto girare. E' ovvio che la prima istruzione eseguita è la prima della riga col numero più basso. Nel nostro caso è proprio la riga 100 che contiene due istruzioni. Subito dopo il computer esamina la riga successiva che, nel nostro esempio, è anche l'ultima. L'utente non si accorge di ciò che il computer sta facendo, ma vede solo il risultato dell'elaborazione che, nel caso specifico, porta alla visualizzazione di:

```
250 2 500
```

Per renderci conto che si può giungere allo stesso risultato per altre vie, impariamo un nuovo comando: NEW (in italiano tnuovo). Digitando NEW ed il tasto di ritorno carrello, il programma memorizzato viene cancellato dalla memoria del calcolatore. Attenzione dunque ad usarlo: un suo utilizzo improprio può annullare ore di lavoro! Nel nostro caso, invece, cancellerà le sole due righe digitate. Ci rendiamo conto che il programma non c'è più digitando LIST, comando già esaminato. Non dovrebbe apparire alcuna riga.

Digitate ora le righe che seguono non tralasciando di battere il tasto Return alla fine di ciascuna di essa:

```
100 A=250               (R)
110 B=2                 (R)
120 PRINT A;            (R)
130 PRINT B;            (R)
140 PRINT A*B           (R)
150 END                 (R)
```

Se ora digitiamo RUN ci accorgiamo che il risultato è perfettamente identico a quello di prima. Ciò dimostra che un risultato può essere raggiunto dalle stesse istruzioni raggruppate, però, nel modo più disparato.

Un nuovo comando (END, che vuol dire fine) è stato aggiunto. Questo, in genere, non è necessario e non ce ne serviremo quasi mai. Nell'eseguire i comandi del programma, infatti, il calcolatore opera fino a che non trova più righe da elaborare. Non appena queste terminano, l'elaborazione si arresta automaticamente.

Il programma appena visto non fa altro che eseguire il prodotto di due numeri. Supponendo di avere in memoria l'ultimo programma esaminato (quello di sei righe) digitate le righe seguenti:

```
100 A=15 (R)
110 B=4 (R)
```

e battete LIST. Dovrebbe apparire il programma modificato:

```
100 A=15
110 B=4
120 PRINT A;
130 PRINT B;
140 PRINT A*B
150 END
```

Ciò dimostra che i contenuti di una linea di programma possono essere modificati digitandoli semplicemente nel modo desiderato e battendo il tasto Return alla fine. Nel caso vogliate cancellare la riga 150, praticamente inutile, battete 150 e poi Return. Chiedendo il listato (LIST) dovrebbe ricomparire lo stesso programma privo, però, della riga in questione.

Siamo dunque riusciti a eseguire, sotto programma, una serie di comandi che sapevamo impartire direttamente seguendo le note riportate nelle prime due puntate. Non dimentichiamo che potremmo conseguire un risultato analogo digitando direttamente i comandi senza ricorrere al programma:

```
A=15:B=2:PRINT A;B;A*B (R)
```

Dov'è allora l'utilità della programmazione?

Non ci si poteva di certo aspettare un miracolo da un programma così semplice!

Purtuttavia dimostreremo che, ricorrendo alla programmazione, è possibile "far di conto" con una semplicità decisamente maggiore che non utilizzando comandi diretti.

Per far ciò è indispensabile imparare un nuovo comando: INPUT (avete letto bene: INPUT e non IMPUT, all'italiana).

Anzitutto, questo comando non può essere impartito direttamente da tastiera, ma solo se giace all'interno di una linea di programma. La sintassi corretta è la seguente:

```
INPUT A
```

Vediamo ora una applicazione pratica. Cancellate il programma presente in memoria (NEW) e assicuratevene digitando LIST. Quindi battete:

```
100 INPUT A
110 PRINT A
```

Esaminiamo l'elementare programmino. Quando diamo il RUN, la riga 100 viene, ovviamente, elaborata per prima. Il calcolatore, però, incontra l'istruzione INPUT e ferma il processo. A questo punto appare sul video un punto interrogativo e il cursore lampeggia al suo fianco. Esso sta a dimostrare che il computer aspetta che noi digitiamo un valore numerico e che battiamo, in seguito, il tasto Return. Accontentiamolo battendo un valore qualunque come, ad esempio:

```
123 (R)
```

Non appena premiamo il tasto di ritorno carrello l'elaborazione prosegue con l'istruzione successiva a INPUT. Poichè sulla riga 100 non ve ne sono altre, viene elaborata la riga 110 che provvede a stampare il valore appena digitato.

Riassumendo: con l'istruzione INPUT noi costringiamo il computer:

1/ a interrompere l'elaborazione;
2/ ad attendere fino a che non digitiamo un valore numerico;
3/ ad associare il valore che digitiamo alla variabile individuata da INPUT ("A" nel caso di INPUT A).

Vediamo ora l'importanza di numerare le righe con intervalli così ampi.

Digitate le righe:

```
90 B=5
110 PRINT A*B
```

Se chiediamo il LIST, dovrebbe apparire:

```
90 B=5
100 INPUT A
110 PRINT A*B
```

Il computer, in altre parole, ha aggiunto al posto giusto la riga 90 (cioè "prima" di 100) e ha sostituito la "vecchia" 110 con la "nuova" 110.

Se diamo il RUN il computer, come di consueto, si fermerà all'istruzione 100 (INPUT) e, dopo che avremo fornito il valore, visualizzerà il valore del prodotto di A*B.

Rendiamo più versatile il programmino. Digitate:

```
90 INPUT B
```

e battete RUN. Il computer, stavolta, fermerà due volte l'elaborazione: la prima per "chiedervi" B e la seconda per A.

Alla fine visualizzerà il prodotto dei due numeri digitati. In effetti il programma cui siamo pervenuti è di certo più versatile (dato che entrambi i fattori possono essere introdotti "sotto" programma) ma non è molto chiaro.

Non appena digitiamo RUN, infatti, appare un punto interrogativo che, specie per chi non conosce il programma, non significa nulla. Rendiamo allora più comprensibile il listato digitando le seguenti linee, prestando la massima attenzione nel battere al posto giusto il carattere di virgolette (") che si ottiene premendo i tasti Shift e "2". Se compaiono caratteri strani, niente paura: premete il tasto Return e ribattete nuovamente la linea dall'inizio.

```
 10 PRINT "ESECUZIONE DI"
 20 PRINT "UN PRODOTTO"
 30 PRINT
 40 PRINT "PRIMO FATTORE"
 95 PRINT "SECONDO FATTORE"
105 PRINT B;"PER";A;"FA";
```

Chiedete il LIST: il computer ha provveduto automaticamente a inserire al posto giusto le varie linee. La loro numerazione è infatti corretta:

10 20 30 40 90 95 100 105 110

Ecco spiegato, dunque, il motivo per cui è bene numerare le linee in modo da consentire successivi, agevoli inserimenti di altre linee di programma.

Digitiamo ora RUN. Se tutto è stato trascritto senza errori, dovrebbe apparire una "cosa" del genere:

ESECUZIONE DI
UN PRODOTTO

PRIMO FATTORE
? 20 (R) (20 lo abbiamo digitato noi)
SECONDO FATTORE
? 30 (R) (30 lo abbiamo digitato noi)
20 PER 30 FA 600

Vediamo ora in che modo si perviene alla visualizzazione descritta.

Quando il programma ha inizio (RUN), viene eseguita la riga col numero più basso (10). Questa non contiene altro che un semplice comando di PRINT seguito, racchiuso da apici, dal messaggio che si desidera far apparire (ESECUZIONE DI).

La riga 20 provvede a visualizzare la seconda parte del messaggio, anch'essa racchiusa da apici (UN PRODOTTO).

La riga 30 chiede di stampare... nulla (=un rigo vuoto) e infatti viene stampato un rigo vuoto con la conseguenza che messaggi successivi appariranno distanziati di un rigo dal messaggio precedente.

La riga 40 fa apparire:

PRIMO FATTORE

La riga 90, che contiene INPUT B, arresta l'elaborazione e fa apparire un punto interrogativo avvertendo in tal modo l'operatore che è necessario digitare un dato per continuare l'elaborazione. Si tenga presente che questa rimane bloccata anche per ore; il calcolatore attenderà pazientemente che venga esaudita la richiesta!

Ora il punto di domanda aiuta l'operatore a rispondere correttamente. Il messaggio precedentemente apparso (PRIMO FATTORE) è piuttosto esplicito e non dovrebbero esserci possibilità di sbagliare. Dopo aver digitato il numero (per esempio: 20) e premuto il tasto Return, l'elaborazione continua con il trattamento della riga 95 che provvede a visualizzare SECONDO FATTORE. Anche in questo caso risponderemo con un valore qualunque (p.e.:30) alla domanda posta dalla riga 100 (INPUT A).

La riga 105 inizia con un PRINT e il calcolatore si predispone per stampare su video. La prima "cosa" che incontra è la variabile B che viene subito visualizzata col valore 30, introdotto, appunto, grazie all'INPUT di riga 90. Poichè dopo di essa c'è un punto e virgola, il contenuto di ciò che si trova tra gli apici (PER) verrà stampato subito dopo il valore di B. A questo punto dell'elaborazione sul video è comparso:

20 PER

Il punto e virgola presente dopo "PER" (riga 105 del programma) obbliga il calcolatore a visualizzare, in coda a PER, il valore della variabile A. Questa vale 30, numero da noi stessi digitato grazie alla riga 100. La parolina "FA" viene visualizzata subito dopo 30 per lo stesso motivo per cui è prima apparso "PER". A questo punto, sul video, possiamo leggere:

20 PER 30 FA

L'ultima "cosa" da stampare è il risultato del prodotto A*B che consente di completare il messaggio e di terminare l'elaborazione:

20 PER 30 FA 600

Digitando nuovamente RUN e fornendo ogni volta valori diversi quando il computer si blocca su INPUT, otterremo in risposta il messaggio "personalizzato".

Provate infatti a digitare le righe seguenti:

```
105 PRINT "MOLTIPLICANDO";A     (R)
110 PRINT "PER";B;               (R)
120 PRINT "SI OTTIENE" A*B       (R)
```

J'esempio visto prima porterebbe, in tal caso, alla visualizzazione di un messaggio più retorico:

MOLTIPLICANDO 20 PER 30
SI OTTIENE 600

E' ovvio che qualsiasi tipo di frase, magari spiritosa, può essere introdotta sostituendo quelle proposte.

Un nuovo comando: GOTO

Impariamo ora un nuovo comando: GOTO che, in inglese, significa "vai a". Quando il computer incontra un'istruzione del genere, "salta" a eseguire la riga numerata col valore presente dopo GOTO.

Se, per esempio, avendo ancora in memoria il programma appena visto, aggiungiamo le righe:

```
150 PRINT: PRINT
200 GOTO 10
```

il programma così modificato, giunto alla riga 200 non interromperà, come nei casi precedenti, l'elaborazione costringendoci a digitare nuovamente RUN, ma continuerà a operare trattando le istruzioni presenti nella linea 10. Questa, guarda caso, è proprio la prima istruzione del nostro programma. In pratica, non appena viene visualizzato il risultato del prodotto, il calcolatore chiederà immediatamente altri dati.

Esempio:

ESECUZIONE DI
UN PRODOTTO

PRIMO FATTORE
?20(R)
SECONDO FATTORE
?30(R)
MOLTIPLICANDO 30
PER 20
SI OTTIENE 600

ESECUZIONE DI
UN PRODOTTO

PRIMO FATTORE
?5(R)
SECONDO FATTORE
?50(R)
MOLTIPLICANDO 50
PER 5
SI OTTIENE 250

eccetera.

Cancelliamo ora il programma spegnendo e riaccendendo il computer e apprestiamoci a digitarne un altro. Il listato conterrà una nuova istruzione che consentirà al calcolatore di effettuare una scelta.

Le scelte del computer

L'istruzione IF...THEN (in italiano: Se...allora), infatti, esamina se (IF) esistono alcune condizioni e, in caso affermativo (THEN), esegue i compiti specificati. In caso contrario l'elaborazione prosegue trattando i comandi contenuti nella riga successiva ignorando le istruzioni poste dopo THEN.

Esempio:

```
100 PRINT "DIGITA UN NUMERO
    QUALUNQUE"
110 INPUT A
120 IF A=0 THEN PRINT
    "UGUALE A ZERO"
130 IF A>0 THEN PRINT
    "MAGGIORE DI ZERO"
140 IF A<0 THEN PRINT
    "MINORE DI ZERO"
```

Diamo il RUN e quando (righe 100-110) appare il messaggio:

DIGITA UN NUMERO
QUALUNQUE

battiamo un numero negativo come, ad esempio, -4.

Vediamo che cosa accade subito dopo aver premuto il tasto Return. L'elaborazione prosegue con la riga successiva a 110, vale a dire la 120. Il calcolatore incontra "IF A=0". Immediatamente esegue un confronto tra 0 e il contenuto di A che, se avete seguito il nostro suggerimento, vale -4, valore, cioè, negativo.

Il confronto porta alla conclusione che "A" non è eguale a zero e, di conseguenza, viene ignorata totalmente la parte della riga 120 successiva a THEN. L'elaborazione salta, quindi, alla riga successiva, la 130.

Un altro confronto deve essere fatto dal computer. Stavolta si tratta di veder se il contenuto di A è maggiore di zero. Nemmeno questa condizione è verificata e anche stavolta ciò che è presente dopo THEN viene totalmente ignorato. L'istruzione della riga 140 riconosce che la condizione (A minore di zero) è vera e, pertanto, vengono eseguite tutte le istruzioni presenti dopo THEN. Nel nostro caso particolare ve ne è solo una che rappresenta l'ordine di stampare il messaggio "MINORE DI ZERO".

Proviamo ora, dopo aver dato nuovamente il RUN, a digitare 34 (o comunque un qualsiasi valore positivo) in risposta a "DIGITA UN NUMERO QUALUNQUE" (riga 100).

Anche in questo caso la condizione della riga 120 (A è eguale a zero?) non si verifica e l'elaborazione passa alla riga successiva. In questa, invece, la condizione viene riconosciuta vera (34 è infatti positivo e dunque maggiore di zero) e l'elaborazione prosegue trattando le istruzioni presenti dopo il THEN. Il messaggio "MAGGIORE DI ZERO" viene visualizzato e l'elaborazione prosegue passando alla linea successiva. Questo fatto è molto importante e porta a risultati indesiderati se preso alla leggera.

Nel nostro caso particolare la riga 140 non riconosce vera la condizione per cui A è minore di zero e, di conseguenza, l'elaborazione termina. Anche introducendo un valore negativo porta a un risultato analogo grazie alla visualizzazione dí "MINORE DI ZERO".

Prima di andare avanti è interessante seguire alcune considerazioni che non solo insegnano qualcosa sulla corretta programmazione, ma evidenziano malfunzionamenti "logici" che possono verificarsi in diversi casi.

Un computer, se mal programmato, può fornire non solo messaggi di errore tipici del computer (come Syntax error, Type mismatch error, eccetera), ma elaborazioni inconcludenti pur se logiche. Ricorriamo a un esempio commentando il seguente programma da digitare dopo aver spento e riacceso il computer:

```
100 PRINT "SCRIVI I DUE
    FATTORI"
110 INPUT A: INPUT B
120 PRINT "IL PRODOTTO DI"
    A "PER" B
130 PRINT "VALE" A/B
```

E' molto simile a quello visto prima.

La riga 100 avverte di ciò che si sta per fare.

La 110 "chiede" i valori dei due fattori ("A" e "B").

La 120 visualizza una frase che però...

...riga 130... è in contraddizione con la 120 e la 100, dato che esegue il calcolo del rapporto tra "A" e "B" (A/B) e non del prodotto di "A" per "B" come dovrebbe (A*B).

Se, in altre parole, al momento di rispondere alle due domande (INPUT) di riga 110 digitassimo "8" e "4", invece del loro prodotto (8x4=32) verrebbe visualizzato il valore "2" fornito dalla divisione di 8 per 4.

Il computer non può aver colpa dell'elaborazione che in definitiva risulta:

IL PRODOTTO DI 8 PER 4
VALE 2

Il vostro computer non ha fatto altro che eseguire gli ordini impartiti e non può certo capire che la parola "PRO-

DOTTO" è in contraddizione col calcolo effettuato (A/B).

Quanto abbiamo esaminato non è altro che un banale esempio di ciò che può capitare in una errata trascrizione di un programma. Questo può, come si dice in gergo, "girare" senza difficoltà alcuna, ma fornisce elaborazioni inconcludenti.

Una nuova funzione: RND(0)

La funzione RND(0) fornisce un numero decimale casuale compreso tra zero e uno. Ecco alcuni di questi valori:

0.456321 0.987536 0.034098 eccetera

Cancellando il programma presente in memoria (mediante il comando NEW, oppure spegnendo e riaccendendo l'apparecchio) provate a digitare il seguente microprogramma. Chi non possiede il computer può, come al solito, seguire attentamente quanto esponiamo:

```
100 A=RND(0): PRINT A: GOTO 100
```

Commentiamolo brevemente.

Alla variabile "A" viene associato un valore casuale, compreso nell'intervallo prima detto.

La seconda istruzione (ricordatevi di inserire **sempre** il carattere di doppio punto ":" tra le varie istruzioni!) fa visualizzare il valore della variabile "A".

L'ultima istruzione (GOTO 100) obbliga il computer a eseguire la riga 100 e, cioè, a ricominciare daccapo.

Se infatti, dopo aver trascritto la riga-programma, battete RUN, vedreste sul video una fila ininterrotta di valori tutti diversi tra loro. Vi ricordiamo che tenendo premuto il tasto CTRL rallenterete la velocità di scorrimento (in inglese: scrolling). Nel caso vogliate fermare il programma è necessario premere il tasto Run/Stop.

Con quest'ultima operazione il vostro Commodore risponderà col messaggio "BREAK IN 100" avvertendovi che l'elaborazione stava eseguendo la riga 100 quando è stato premuto il tasto di interruzione di programma. Questo messaggio sarà utilissimo, in programmi lunghi, allo scopo di rintracciare la riga in corso di elaborazione.

Tornando al programma di prima, vediamo di modificarlo nel modo seguente:

```
100 A=RND(0)*10: A=INT(A)
110 PRINT A: GOTO 100
```

Dovrebbe essere inutile ricordare che la digitazione della nuova riga 100 cancella automaticamente la riga precedente numerata con lo stesso valore.

Esaminiamo le quattro istruzioni del nuovo programma.

Riga 100: il valore casuale RND(0) viene moltiplicato per 10 allo scopo di avere un numero decimale compreso tra zero e nove. Per evitare lunghe e noiose spiegazioni il lettore può controllare, nella tabella che segue, la differenza esistente tra il valore casuale (colonna di numeri di sinistra) e lo stesso valore moltiplicato per 10 (colonna centrale).

La seconda istruzione A=INT(A) di riga 100 "associa" alla stessa variabile "A" il suo valore intero, vale a dire che il comando INT "tronca" l'eventuale parte decimale di un qualsiasi valore numerico. Quando la riga 100 è stata eseguita, in "A" sarà presente un valore intero compreso inevitabilmente tra zero e nove (vedi colonna di destra nella tabella).

RND(0)	A=RND(0) *10	A= INT(A)
0.14567	1.4567	1
0.67237	6.7237	6
0.93451	9.3451	9
0.01909	0.1909	0

La riga 110 visualizza il valore in oggetto e (GOTO 100) ricomincia il calcolo.

Approfittiamo di questa occasione per chiarire meglio alcuni casi che interessano il segno eguale (=).

In effetti lo abbiamo finora usato molto spesso e non sarà innopportuno ricordare qualche esempio:

A=123: Bt -7.72

Nei casi appena visti, alla variabile "A" sarà associato il valore 123, alla "B" -7.72 e così via.

C'è però una sottile differenza tra ciò che abbiamo imparato a scuola e ciò che viene interpretato dal computer. Siamo infatti abituati al seguente ragionamento: se "X" vale 100 e "Y" vale 200, con l'espressione Z=X+Y evidentemente "Z" assumerà il valore di 300.

Se però adottiamo lo stesso ragionamento nella espressione seguente:

X=X+10

giungiamo alla conclusione:

100=100+10

e cioè alla vistosa contraddizione:

100=110

Il numero 100 non può contemporaneamente valere anche 110.

Per il computer, però, non vi è alcuna contraddizione nell'espressione precedente perchè interviene il fattore "tempo".

Spieghiamoci meglio: l'espressione X=X+10 viene interpretata dal computer nel modo seguente: alla variabile "X" associa **adesso** il valore fornito dalla somma di 10 più il valore che X valeva **prima** della presente istruzione".

Se, in altre parole, "X" valeva 20, dopo l'esecuzione di X=X+10 varrà 30.

Nel programma visto prima, l'istruzione A=INT(A) associerà **ora** alla variabile "A" il valore intero del numero decimale che "A" aveva **prima** di incontrare l'istruzione stessa.

Il tuo primo giochino

Cancelliamo il programma di due righe appena esaminato e commentiamo il seguente:

```
 80 A1$="E' TROPPO PICCOLO"
 90 A2$="E' TROPPO GRANDE"
100 A=RND(0)*10: A=INT(A)
```

```
110 PRINT "INDOVINA
    UN NUMERO"
120 PRINT "COMPRESO TRA 0 E 9"
130 INPUT B
140 IF B<A THEN PRINT B;A1$:
    GOTO 130
150 IF B>A THEN PRINT B;A2$:
    GOTO 130
160 PRINT "INDOVINATO!":
    GOTO 100
```

Il listato rappresenta un gioco che consiste nell'indovinare un numero intero compreso tra zero e nove.

Esaminiamolo nei dettagli.

Riga 80: viene definita la stringa alfabetica A1$ e a essa viene associata la frase "E' TROPPO PICCOLO".

Analogamente la riga 90 definisce, come A2$, la frase "E' TROPPO GRANDE". Vedremo tra poco come utilizzeremo le due variabili stringa.

La riga 100 memorizza nella variabile "A", come già visto, un valore intero casuale compreso tra "0" e "9".

Le righe 110 e 120 visualizzano un breve messaggio che non è determinante ai fini del gioco, ma conferisce una certa "estetica" allo stesso.

La riga 130 "chiede" (INPUT) il valore che digiterete.

Ora viene il bello. Supponiamo che il computer abbia associato ad "A" il valore casuale 3 grazie alle istruzioni della riga 100.

Quando appare il punto interrogativo tipico dell'INPUT di riga 130, se si digita 2 (e, ovviamente, si preme Return) l'elaboratore proseguirà con la riga successiva a 130 e cioè 140. A causa dell'istruzione IF...THEN deve fare un confronto tra i valori di "A" e di "B".

Poichè, nel caso ipotizzato, B è effettivamente minore di A, verranno eseguite tutte le istruzioni successive alla parola THEN.

La prima di queste è:

```
PRINT B;A1$
```

che visualizza il valore B seguito subito dopo (notare il carattere di punto e virgola ";") dalla stringa A1$. Questa contiene (esaminate la riga 80) una certa frase e il risultato di PRINT B;A1$ sarà in definitiva:

```
2 E' TROPPO PICCOLO
```

La seconda (e ultima) istruzione che si incontra dopo THEN di riga 140 è GOTO 130, che ci permetterà di continuare il gioco impostando un altro valore. Supponiamo ora di digitare "5" alla nuova apparizione del punto di domanda di riga 130 e, dovreste averlo imparato, il tasto Return.

Anche in questo caso l'elaborazione prosegue con la riga 140 ma, poichè il valore della variabile "B" (cioè 5) non è minore di "A" che vale ancora 4, tutto ciò che è presente dopo THEN non verrà nemmeno preso in considerazione e il calcolatore continuerà con la riga successiva a quella contenente IF...THEN.

Si giunge così alla riga 150 nella quale l'espressione IF B<A è verificata e di conseguenza, con ragionamenti analoghi a quelli visti prima, viene visualizzato il messaggio:

```
5 E' TROPPO GRANDE
```

Anche in questo caso GOTO 130 farà proseguire il programma dalla riga 130 e sarà chiesto (INPUT) un nuovo valore.

A questo punto, se 5 è troppo grande e 3 troppo piccolo, digiteremo 4 e batteremo Return. La riga 140 sarà scartata e così pure la 150, dato che non si verificano le diseguaglianze ivi indicate.

Si perviene, di conseguenza, alla riga 160 che visualizza "INDOVINATO!" e fa ripartire il programma non più dalla riga 130 ma dalla 100.

Se infatti il comando presente nella riga 160 fosse GOTO 130, come in quelle esaminate precedentemente, il programma continuerebbe lasciando inalterato il contenuto della variabile A che avrebbe, così, lo stesso valore precedente (54) dato che non è intervenuto nessun "fatto" nuovo ad alterare il contenuto di detta variabile.

Con l'istruzione GOTO 100, invece, si utilizza nuovamente RND(0) che modifica il valore precedente di "A". Attenzione: il caso potrebbe associare nuovamente lo stesso valore 4 alla variabile "A".

Vediamo ora di interrompere il programma per inserire altre istruzioni. Se avete un C-16 (o Plus-4), premete il tasto Commodore (il primo in basso a sinistra) insieme col tasto Run/stop.

Con un C-64, C-128 oppure Vic-20 se provate a premere il tasto Run/stop mentre il computer attende che battiate un valore (istruzione INPUT), non otterrete l'effetto di interruzione. Per interrompere un programma "bloccato" sull'istruzione INPUT è necessario, tenendo schiacciato il tasto Run/stop, battere anche il tasto Restore: lo schermo si cancella e compare la scritta READY. Non preoccupatevi: il programma è ancora allocato in memoria. Sinceratevene battendo LIST e il tasto Return.

Volete ora rendere più difficile il gioco? Apportate la semplice modifica:

```
100 A=RND(0)*100: A=INT(A)
120 PRINT "COMPRESO TRA 0 E
99"
```

In questo caso non avrete più 10 numeri possibili (da 0 a 9) come prima, ma 100 (da 0 a 99).

Vedremo ora di rendere più completo il programma dimostrando nel contempo che è piuttosto semplice aggiungere istruzioni.

Poichè abbiamo apportato parecchie modifiche e ne dovremo aggiungere altre, è meglio, onde evitare confusione nel lettore, riportare il listato completo prima di commentarlo:

```
 80 A1$= "E' TROPPO PICCOLO"
 90 A2$= "E' TROPPO GRANDE"
 95 I=0
100 A=RND(0)*100: A=INT(A)
110 PRINT "INDOVINA
    UN NUMERO"
120 PRINT "COMPRESO TRA 0 E
99"
130 INPUT B
135 I=I+1
14  IF B<A THEN PRINT B:A1$:
```

```
    GOTO 130
150 IF B>A THEN PRINT B;A2$:
    GOTO 130
155 PRINT"TENTATIVI N.";I
160 PRINT "INDOVINATO!":
    GOTO 100
```

Le righe aggiunte, rispetto al precedente programma, sono la 95, la 135 e la 155.

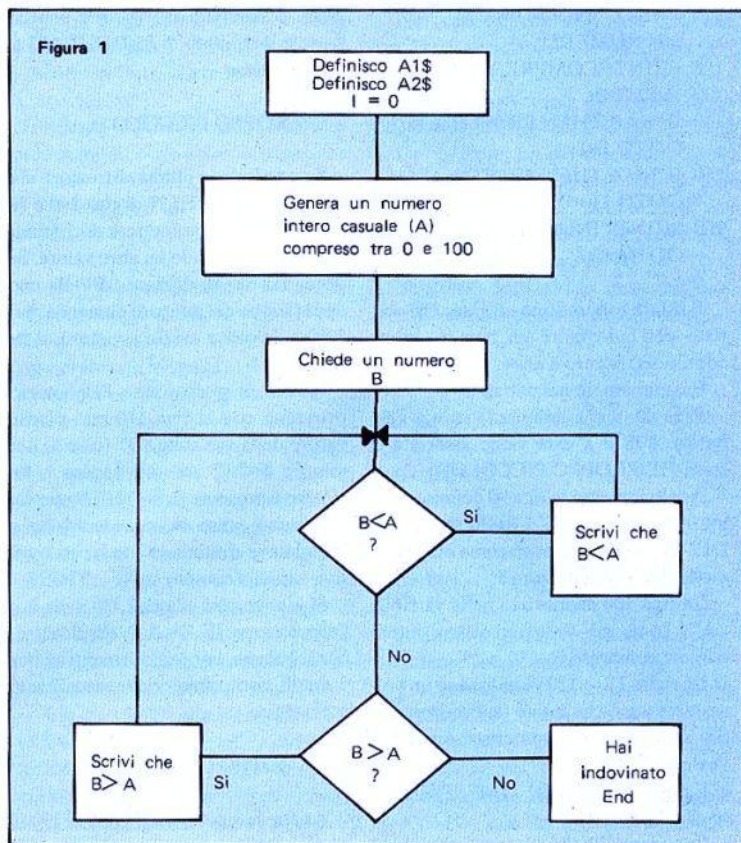
Con I=0 mettiamo a zero (o, come si suol dire, resettiamo) la variabile "I". La riga 135 incrementa il valore di "I". Ciò significa che se prima "I" valeva zero, dopo l'esecuzione dell'istruzione varrà "1" oppure se prima valeva 13 dopo varrà 14 e così via.

Si noti che l'incremento di "I" viene effettuato tutte le volte che rispondiamo alla "domanda" INPUT di riga 130 indipendentemente da ciò che battiamo. Abbiamo cioè realizzato un "contatore di tentativi".

Quando al termine del gioco indoviniamo il valore del computer, verrà visualizzato (PRINT I di riga 155) il numero di tentativi.

Si noti l'importanza di resettare (porre a zero) la variabile "I" alla riga 95. Se questa istruzione fosse assente, potrebbe verificarsi un inconveniente.

Supponiamo infatti di aver indovinato il numero al settimo tentativo. In questo caso "I" assume il valore 7 e l'elaborazione (cfr riga 160) continuerebbe alla riga 100 conservando inalterato il valore della variabile "I". Questo continuerebbe ad essere incrementato (riga 135) a ogni successivo tentativo effettuato per indovinare il nuovo numero. Se, ad esempio, lo indoviniamo dopo 4 tentativi, il computer, grazie alla riga 155, visualizzerà "TENTATIVI N.11" considerando anche i tentativi della manche precedente.

Ciò dimostra che, pur se privo di errori di sintassi, un programma mal progettato può portare a incongruenze. Provate, tanto per dimostrare ulteriormente quanto asserito, a scambiare tra loro A1$ e A2$ nelle righe 140 e 150. Se il valore digitato sarà troppo grande, apparirà il messaggio "TROPPO PICCOLO" e viceversa, mandando nel pallone l'ignaro giocatore!

Riportiamo in figura 1 il diagramma di flusso, vale a dire il percorso "logico" che viene seguito sul programma presentato.

Il ciclo FOR...NEXT

Siamo così giunti alla descrizione di un'altra forma sintattica accettata dal computer, vale a dire una forma che consente di ripetere un ciclo per un numero prefissato di volte.

Ricorriamo ad un esempio da digitare dopo aver spento e riacceso il calcolatore:

```
100 FOR I=1 TO 10
110 PRINT I
120 NEXT I
```

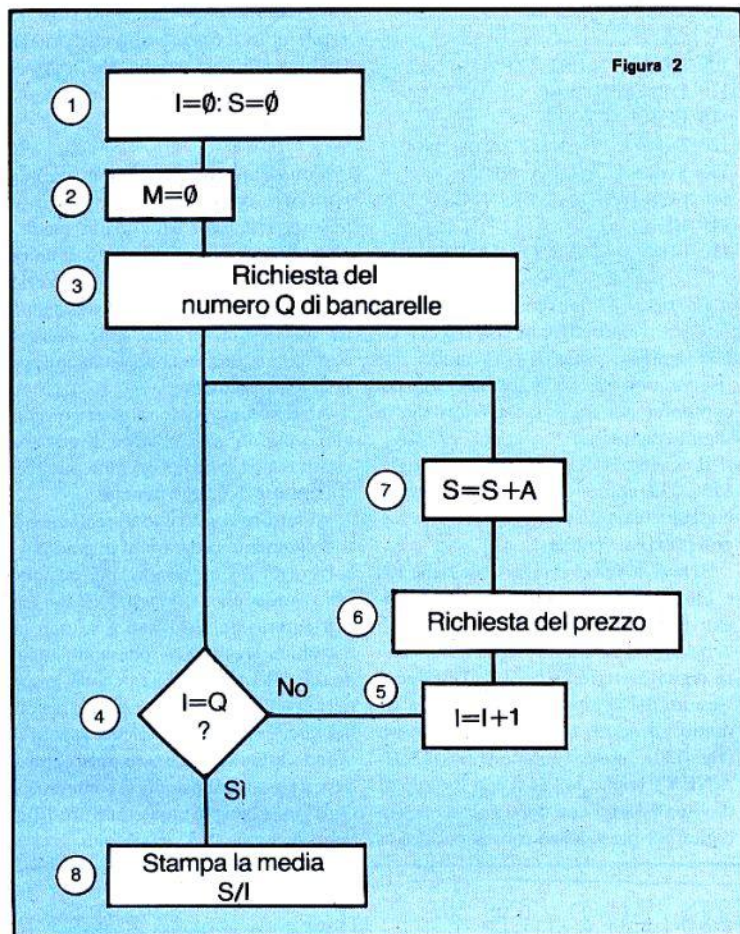
Proviamo a tradurre letteralmente le tre istruzioni.

"Per I che va da 1 fino a 10 (FOR I=1 TO 10) stampa il valore di I (PRINT I). Continua il conteggio (NEXT I).

In altre parole, verranno eseguite dieci volte (It1 TO 10) tutte le istruzioni (nel nostro caso una sola: PRINT I, cioè visualizza il valore di "I") comprese tra la parola FOR e la parola NEXT.

In effetti NEXT ha il solo scopo di individuare la "chiusura" del ciclo da eseguire (nel nostro caso 10 volte).

Apportate le seguenti modifiche:

```
100 FOR I=1 TO 10
110 PRINT"QUANDO I VALE":I
120 PRINT "IL QUADRATO
```

```
VALE";I*I
140 NEXT I
```

In quest'ultimo caso verrà visualizzato dieci volte il valore di "I" preceduto dal messaggio "ORA I VALE" e seguito dal calcolo del suo quadrato (I*I di riga 130) con relativo messaggio.

E' ovvio che il computer, ogni volta che giungerà alla riga 140, incrementerà il valore di "I" e riprenderà automaticamente l'elaborazione a partire dalla riga 110. Non appena "I" varrà 10, il ciclo si interrompe e il programma termina. Lo stesso risultato poteva esser raggiunto ricorrendo a molti altri sistemi o utilizzando diversi messaggi.

Un'applicazione pratica del ciclo FOR...NEXT

In questo paragrafo vedremo in che modo, potendo partire da un problema concreto, sia possibile giungere a una soluzione con l'aiuto del computer.

E' ovvio che ricorreremo a un esempio semplicissimo che, risolto dapprima in modo banale, sarà sempre più sofisticato con varie aggiunte o modifiche.

Problema: "Al mercato comunale una certa qualità di frutta viene posta in vendita a prezzi diversi sulle varie bancarelle. Calcolare il prezzo medio di vendita".

E' noto che la media di N elementi è data dalla somma degli N elementi stessi divisa per N. Per esempio, la media tra i valori 1200, 1600, 1500, 1500 (=quattro elementi) è data da:

M= (1200+1500+1400+1500)/4=
= 5600/4= 1400

Un diagramma di flusso può essere quello mostrato in figura 2.

Commentiamolo simbolo per simbolo:

1-2- Alla "partenza" del programma le variabili "I" ed "M" sono resettate (poste, cioè a zero). Ad esse verrà affidato il compito di memorizzare il numero di prezzi esaminati (I) e la loro somma (S).

3 - Viene chiesta la quantità (Q) di bancarelle che verrà presa in considerazione. Alla variabile "Q" sarà associato tale valore allo scopo di confrontarlo (vedi parte 4) volta per volta con la quantità di prezzi introdotti.

4 - Viene effettuato un confronto tra i contenuti delle variabili "A" e "Q".

5 - Se il confronto del punto 4 è negativo, (vale a dire se non sono stati introdotti tutti i prezzi di vendita al dettaglio), la variabile "I" viene incrementata di un'unità per ricordare che sta per essere introdotto un nuovo prezzo.

6 - Viene richiesto il prezzo di vendita al pubblico dell'I-esima bancarella.

7 - Alla variabile "S" (che contiene la somma dei prezzi esaminati nelle fasi precedenti), viene aggiunto il valore digitato nella fase 6. Subito dopo il programma riprende dalla fase 4.

Nuovamente viene effettuato un confronto tra tra "I" e "Q" per controllare se tutti i prezzi sono stati introdotti.

In caso negativo il ciclo si ripete (fasi 5, 6, 7, 4); in caso positivo si passa alla fase 8.

8 - Viene stampato il risultato di S/I. A questo punto delle operazioni, infatti, "S" contiene la somma dei vari prezzi introdotti uno alla volta (per "I" volte), mentre ad "I" è associato, appunto, il numero degli stessi valori. Il rapporto S/I rappresenta quindi la media aritmetica dei valori introdotti.

Una delle "traduzioni" in Basic che suggeriamo è la seguente:

```
100 PRINT "N. BANCARELLE":
    INPUT Q
110 I=0: S=0
120 I=I+1:PRINT I;"PREZZO":
    INPUT A
130 S=S+A: IF I<Q THEN 120
140 PRINT"PREZZO MEDIO";S/Q
```

Utilizzando, invece, il ciclo FOR-...NEXT, ecco una versione dello stesso programma:

```
100 PRINT"N.PREZZI": INPUT Q
110 S=0: FOR I=1 TO Q
120 PRINT I;"PREZZO": INPUT A
130 S=S+A: NEXT
140 PRINT"PREZZO MEDIO";S/Q
```

Il risultato potrebbe, però, generare qualche dubbio. Come facciamo, infatti, a controllare che i valori introdotti sono stati digitati senza errori? Sarebbe opportuno un controllo finale per verificare che i vari prezzi siano quelli "giusti". Un modo di memorizzare i singoli prezzi di vendita può esser quello di inserirli in un vettore, vale a dire in una variabile a indice come quelle incontrate nella prime due puntate.

Ecco un programma più complicato:

```
100 INPUT"N.PREZZI";Q
110 DIM A(Q): S=0
115 FOR I=1 TO Q
120 INPUT"PREZZO";A: A(I)=A
130 S=S+A: A(I)=A: NEXT
140 FOR I=1 TO Q: PRINT A(I)
145 NEXT
150 PRINT"PREZZO MEDIO";S/Q
```

La riga 110 "dimensiona" il vettore A(Q) al valore digitato con INPUT Q. Ciò significa, in parole più semplici, che riserva uno spazio nella memoria del computer per registrare i valori che in seguito digiteremo.

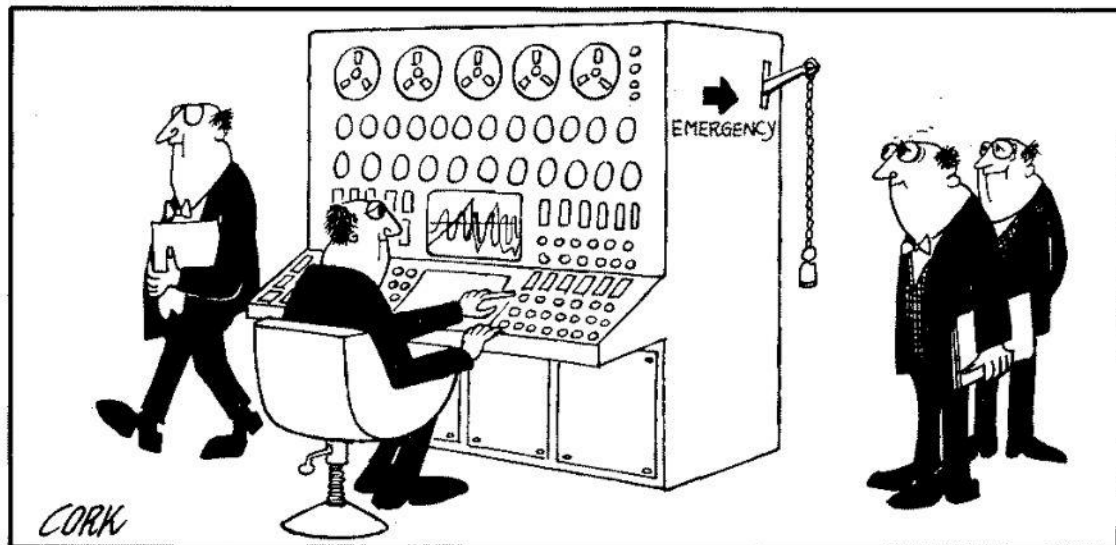
Il ciclo FOR...NEXT delle righe 115...130 consente di memorizzare in ciascun elemento del vettore A(X) i singoli prezzi di vendita.

Si noti la nuova sintassi delle righe 100 e 130: faranno apparire il punto interrogativo (tipico dell'INPUT) subito dopo la parola racchiusa tra virgolette. Il valore digitato, verrà associato all'elemento I-esimo del vettore "A". Quando si saranno introdotti tutti i valori desiderati (riga 130), inizierà un nuovo ciclo FOR-...NEXT (righe 140 145) con il compito di visualizzare l'uno dopo l'altro i prezzi digitati in precedenza consentendone la verifica. L'ultima istruzione (riga 150) rende noto il calcolo S/Q che visualizza (finalmente!) la media dei prezzi di vendita.

Conclusioni

Con gli argomenti trattati in queste tre puntate e con i suggerimenti indicati, il lettore avrà senz'altro avuto modo di constatare la semplicità d'uso di un personal computer. Non tutte le istruzioni del linguaggio Basic sono state esaminate, ma questa carenza, lungi dall'esser considerata una lacuna, è stata introdotta intenzionalmente.

Altri volumi possono esser consultati allo scopo di approfondire diversi argomenti e altri articoli verranno pubblicati dall'autore di queste puntate.

Al lettore auguriamo di realizzare autonomamente programmi in grado di risolvere problemi sempre più complessi utilizzando uno strumento, il personal computer, che altro non è se non una macchina realizzata pensando unicamente alle meravigliose facoltà proprie del cervello umano: la logica e la razionalità che, considerando alcuni aspetti del mondo attuale, sembrano purtroppo cedere il passo, nonostante le conferenze di Ginevra, a ben più inquietanti problematiche decisamente... irrazionali.

computer service

VENDITA PER CORRISPONDENZA

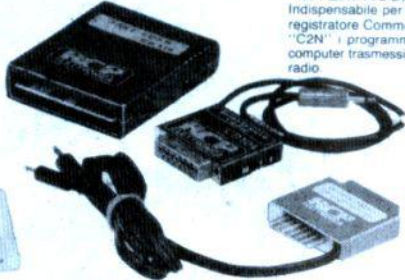
ACCESSORI PER COMPUTER COMMODORE

GRUPPO CONTINUITÀ
Fornito senza le 12 batterie a stilo ricaricabili. Consente il funzionamento del Vostro computer Commodore C64 o VIC 20 in assenza di corrente. Durata di funzionamento 30 minuti. Ricarica tramite alimentatore Commodore.

KIT ALLINEAMENTO TESTINA
Composto dal cacciavite, nastro di controllo e strumento di taratura con monitor audio permette il perfetto allineamento dei registratori digitali anche con nastri commerciali.

VELOCIZZATORE DI CARICAMENTO FLOPPY
Cartridge con un insieme di utility residenti su ros per velocizzare il drive nel Commodore 64.

INTERFACCIA RADIO
Indispensabile per registrare con registratore Commodore modello "C2N" i programmi speciali per computer trasmessi dalle emittenti radio.

CUFFIA PER COMMODORE C 64
Leggerissima permette l'ascolto personale del computer evitando di disturbare durante i giochi.

COPIATORE PROGRAMMI
Dispositivo hardware per effettuare copie di nastri protetti o turbo utilizzando due registratori Commodore o compatibili.

DUPLICATORE CASSETTE
Indispensabile per realizzare delle copie, con un registratore normale, di un nastro protetto o con caricamento turbo.

Articolo	Codice	Prezzo
Bus quadrislot	Art. CD 100	L. 55.000
Interfaccia cassette	Art. CD 101	L. 30.000
Duplicatore cassette	**Art. CD 102**	**L. 30.000**
Copiatore programmi	**Art. CD 103**	**L. 30.000**
Interfaccia radio	**Art. CD 104**	**L. 30.000**
Kit allineamento testina	**Art. CD 105**	**L. 47.000**
Alimentatore per C64 e VIC 20	Art. CD 106	L. 45.000
Gruppo continuità (fornito senza le 12 batterie a stilo ricaricabili)	**Art. CD 107**	**L. 66.000**
Pacco batterie (12 stilo 1,2 Volt ricaricabili	Art. CD 117	L. 52.000
Commutatore antenna TV/computer	Art. CD 108	L. 9.500
Tasto reset	Art. CD 109	L. 5.500
Interfaccia Centronics	Art. CD 112	L. 104.000
Espansione di memoria per C 16	Art. CD 114	L. 158.000
Velocizzatore di caricamento flop.	**Art. CD 115**	**L. 49.000**
Espansione di memoria per VIC 20 16K	Art. CD 116	L. 112.000
Modulatore Executive	Art. CD 120	L. 72.000
Penna ottica grafica	Art. CD 121	L. 45.000
Tavoletta grafica	Art. CD 130	L. 238.000
Multipresa con filtro - 2 prese	Art. CD 140	L. 41.000
Cuffia per Commodore C 64	**Art. CD 150**	**L. 19.000**
Stabilizzatore elettronico di tensione 500 W	Art. CD 160	L. 430.000
Gruppo di cintinuità 60 W	Art. CD 170	L. 400.000
Gruppo di continuità 200 V	Art. CD 180	L. 802.000
Inventer 12 Volt cc. 220 Volt ca. 100 Watt	Art. CD 190	L. 297.000
Cavo alimentazione	Art. CD 200	L. 4.600
Cavo drive o stampante Commodore	Art. CD 205	L. 8.500
Prolunga per Joystick - mt. 3	Art. CD 210	L. 25.000
Prolunga per cavo TV - mt. 3	Art. CD 215	L. 12.500
Cavo audio - mt. 6	Art. CD 220	L. 15.500
Adattatore Joistick (Atari e C64 al C 16)	Art. CD 225	L. 10.500
Adattatore registratore per C 16	Art. CD 226	L. 19.500
Mascherina antiriflesso 12"	Art. CD 300	L. 35.000
Nastro inchiostrato per Tally - mt. 80	Art. CD 610	L. 16.500
Nastro inchiostrato per Tally - mt. 180	Art. CD 611	L. 16.500
Nastro inchiostrato per Tally 1000 e Honeywell	Art. CD 612	L. 9.500
Nastro inchiostrato per Commodore MRS 801	Art. CD 614	L. 13.000
Nastro inchiostrato per Commodore MPS 802	Art. CD 616	L. 18.000
Nastro inchiostrato per Commodore MPS 803	Art. CD 618	L. 19.500
Mause per Commodore C 64	Art. CD 860	L. 240.000
Pacco carta lettura facilitata 24"×11" modulo da 500 fogli con bordi a strappo	Art. CD 630	L. 13.500
Supporto stampante porta carta in plexiglass "fume" - normale	Art. CD 660	L. 59.000
Supporto stampante porta carta in plexiglass "fume" - rinforzato	Art. CD 670	L. 80.000
Floppy disk 5" singola faccia doppia densità "ODP" - conf. 10 pezzi	Art. CD 700	L. 40.000
Floppy disk 5" singola faccia doppia densità "CBS" - conf. 10 pezzi	Art. CD 702	L. 38.000
Floppy disk 5" singola faccia doppia densità "VERBATIM" - conf. 10 pezzi	Art. CD 704	L. 42.000
Floppy disk 5" singola faccia doppia densità "DYSAN" - conf. 10 pezzi	Art. CD 706	L. 68.000
Nastri magnetici C 10 digitali - conf. 10 pezzi	Art. CD 712	L. 20.000
Nastri magnetici C 15 digitali	Art. CD 714	L. 21.000
Copritastiera in plexiglass per C64 - C16 e VIC 20	Art. CD 750	L. 16.000
Copritastiera in stoffa per C64 - C16 e VIC 20	Art. CD 760	L. 10.500
Vaschetta portafloppy in plexiglass per 40 dischi con chiave	Art. CD 770	L. 30.000
Vaschetta portafloppy in plexiglass per 90 dischi con chiave	Art. CD 780	L. 37.000
Kit pulizia testine registratore	Art. CD 815	L. 13.500
Kit pulizia disk drive	Art. CD 820	L. 26.000
Kit pulizia tastiera	Art. CD 830	L. 16.500
Foratore disk in plastica (per utilizzare la seconda faccia dei dischi)	Art. CD 840	L. 10.000
Foratore disk in metallo "tako"	Art. CD 849	L. 14.000
Joystick Spectravideo II	Art. CD 850	L. 27.000
Joystick.a Microswitch	Art. CD 851	L. 52.500
Joystick senza fili con unità ricevente (funziona a batteria)	Art. CD 852	L. 98.000
Joystick per Commodore 16 (originale)	Art. CD 130	L. 29.500

TUTTI I PREZZI SONO COMPRENSIVI DI IVA
NON SI ACCETTANO ORDINI INFERIORI A L. 30.000
CONTRIBUTO FISSO SPESE DI SPEDIZIONE L. 5000

SI ACCETTANO ANCHE ORDINI TELEFONICI AI NUMERI 0522/661647-661471

BUONO DI ORDINAZIONE

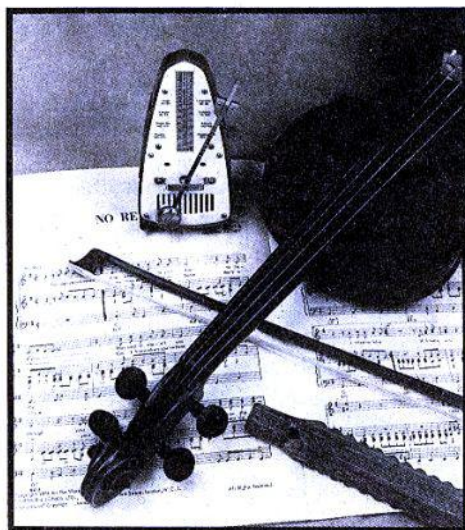
NOME - COGNOME

INDIRIZZO   N.

C.A.P.   CITTÀ   PROVINCIA

VOGLIATE INVIARMI IN CONTRASSEGNO

N.	Art.	L.
N.	Art.	L.
N.	Art.	L.
SPESE SPEDIZIONE		L. 5.000
PAGHERÒ AL POSTINO		L.

COMPUTER SERVICE VIA A. MANZONI, 49 - 42017 NOVELLARA (RE) - TEL. (0522) 661647

Incontri ravvicinati di tipo musicale

di **D.R.Matturro & M.L.Nitti**

Un utilissimo programma grafico-musicale per usare correttamente i parametri fondamentali per la musica col C-64, ossia l'attack decay sustain release.

Tutte le volte che si parla di musica per il Commodore 64 si finisce irrimediabilmente per proporre programmi per la creazione di composizioni musicali, di effetti sonori o per la modifica dei parametri del sintetizzatore trascurando di desciverne minuziosamente il funzionamento.

Questa volta, invece, saranno proprio le nozioni fondamentali a fare la parte del leone. Se approfondite, queste permetteranno finalmente una piena comprensione del "fenomeno suono" secondo uno degli aspetti che maggiormente lo caratterizza.

Proprio per non generalizzare ci limiteremo ad una panoramica sulle caratteristiche del SID; questo è appunto il nome del generatore sonoro elettronico di cui è munito il nostro caro 64.

In primo luogo, per chi ancora non se ne fosse accorto, bisogna constatare che non esistono vere e proprie istruzioni musicali e che bisogna sopperire a questa carenza (purtroppo) facendo uso delle istruzioni PEEK e POKE. Ne consegue che esistono locazioni di memoria opportunamente predisposte a contenere i valori relativi ai parametri del suono. Tali locazioni, per l'esattezza 28, si trovano nelle zone di memoria RAM numerate da 54272 a 54300.

Una rapida occhiata alle caratteristiche del SID fornirà alcuni chiarimenti. Questo generatore sonoro possiede tre voci, o meglio tre oscillatori, un controllo di volume generale e molte altre diavolerie che non prendiamo nemmeno in considerazione per non rovinarci subito l'esistenza. Basta ricordare che tali "diavolerie" possono permettere la produzione di superbe alchimie e che queste, come tali, possono essere architettate solo da esperti alchimisti quali per ora non siamo (ma certamente lo diventeremo).

Una prima regola, facile faciile

Il controllo del volume generale, cioè di tutte e tre le voci, risiede nella locazione 54296; per settarlo (tregolarlo, impostarlo) al massimo dobbiamo semplicemente utilizzare l'istruzione:

```
POKE 54296,15
```

Prima di sperimentare qualsiasi "giochetto" musicale, dobbiamo quindi "alzare" il volume tenendo presente che 15 è appunto il valore massimo (partendo da zero).

Le locazioni delle "voci"

La tabella dei registri (=locazioni RAM) del SID ci dice come sono organizzate le 28 locazioni che gli appartengono; tralasciando quelle che vanno dalla 54293 alla 54300 (riservate alla maggior parte delle alchimie di cui sopra) rimangono tre gruppi di locazioni (tabella A), relativi alle tre voci, organizzati allo stesso modo:

54272-54278	voce 1
54279-54285	voce 2
54286-54292	voce 3

Sette locazioni per voce quindi. Per realizzare un primo e sostanziale approccio alla musica del 64 basta dunque usare solo sette locazioni ed è infatti quello che noi faremo. Le celle di memoria dalla 54272 alla 54278 sovraintendono alla voce 1; per comodità assegneremo ad una variabile tale valore in modo da gestire più velocemente tutti gli indirizzi:

```
S=54272
```

Delle sette locazioni, le prime due (S,S+1, cioè 54272 e 54273) sono usate rispettivamente per il byte basso ed il byte alto della nota da suonare; questo significa che dovremo "pokkare" in queste i valori relativi alla frequenza della nota scelta.

Ahi, ahi; ecco il primo termine tecnico poco digeribile. La frequenza è un parametro importante del suono; ogni nota

TABELLA PER LA LETTURA DELLE SIGLE		
VMAX	-	LIVELLO VOLUME MASSIMO
L.S.	-	LIVELLO DEL SUSTAIN
T.A.	-	TEMPO DI ATTACK
T.D.	-	TEMPO DI DECAY
T.R.	-	TEMPO DI RELEASE
P.R.	-	PUNTO DI RELEASE

vibra un certo numero di volte al secondo, le note più basse vibrano poco, al contrario di quelle alte. Ogni nota quindi ha una sua frequenza che si misura in Hertz.

Una maggiore altezza (non è il volume) della nota, dunque, corrisponderà ad un maggiore numero di hertz: questo ci dice che la nota DO più grave avrà differenti vibrazioni di un DO più acuto, anche se si tratta della stessa nota.

I valori relativi alla frequenza sono reperibili in qualsiasi manuale Commodore e quindi, per lasciare spazio a cose più importanti, preferiamo evitare di trascriverli in questa sede.

Riportiamo invece una tabella di riferimento (Tabella A) per associare con facilità alle locazioni dei registri del SID il loro utilizzo. Sempre rimanendo in tema di note, bisogna tenere presente che l'altezza di cui sopra si definisce in ottave; un MI dell'ottava 1 è molto più basso e grave di un acuto MI dell'ottava 7.

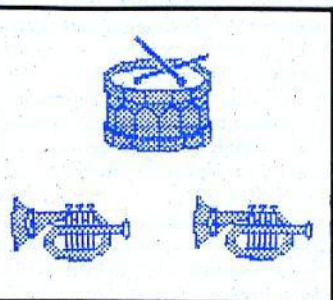
Ma torniamo alle locazioni della voce uno. Esemplificando, per suonare il LA dell'ottava 4 (frequenza 440 Hertz) bisogna fare quanto segue:

POKE S,28 Byte basso della frequenza
POKE S+1,214
Byte alto della frequenza
I valori sono quelli della tabella delle note nel manuale della Commodore e le locazioni si riferiscono alla voce 1 che abbiamo preso in esame.

La forma d'onda

Continuando nell'esame delle locazioni, saltando le 54274-54275 su cui torneremo dopo, incontriamo la 54276 adibita alla selezione della forma d'onda, ovvero

TABELLA A REGISTRI DEL SID

INDIRIZZO	DESCRIZIONE
54272	BYTE BASSO DELLA FREQ.
54273	BYTE ALTO
54274	BYTE BASSO PULSE
54275	BYTE ALTO
54276	FORMA D'ONDA
54277	ATTACK/DECAY
54278	SUSTAIN/RELEASE
** VOCE 2 **	
54279	BYTE BASSO DELLA FREQ.
54280	BYTE ALTO
54281	BYTE BASSO PULSE
54282	BYTE ALTO
54283	FORMA D'ONDA
54284	ATTACK/DECAY
54285	SUSTAIN/RELEASE
** VOCE 3 **	
54286	BYTE BASSO DELLA FREQ.
54287	BYTE ALTO
54288	BYTE BASSO PULSE
54289	BYTE ALTO
54290	FORMA D'ONDA
54291	ATTACK/DECAY
54292	SUSTAIN/RELEASE
** ALTRI **	
54293	FREQUENZA BASSA FILTRO
54294	FREQUENZA ALTA
54295	RISONANZA
54296	VOLUME GENERALE
** REGISTRI A SOLA LETTURA **	
54297	VALORE PADDLES
54298	VALORE PADDLES
54299	STATUS OSCILLATORE 3
54300	STATUS INVILUPPO 3

del tipo di oscillazione. La forma d'onda conferisce un "colore" particolare al timbro, ma non è questa la sede per trattarla approfonditamente. E' necessario tuttavia sapere che ci sono 4 possibilità nel SID e corrispondono a 3 oscillazoni diverse più una "variabile". Queste si settano semplicemente "pokkando" uno dei seguenti valori nella locazione 54276:

Valore	Forma d'onda
POKE S+4,17	Triangolare
POKE S+4,33	Dente di sega
POKE S+4,129	Rumore

L'oscillazione variabile, che viene denominata PULSE, può essere settata così:

POKE S+4,65 Pulse

Ma questa operazione non basta; la forma d'onda emessa deve anche essere controllata mediante l'uso delle due locazioni precedentemente "saltate" (54274-54275). In base ai valori inseriti in questi due indirizzi, si producono oscillazioni, e quindi timbriche, diverse. Per il momento basta precisare che un'onda quadra è ottenibile selezionando in questo modo:

POKE S+2,0
POKE S+3,8
POKE S+4,65

Il controllo dell'A.D.S.R.

Analizzate le prime cinque locazioni, ne rimangono due; proprio queste due sono quelle che maggiormente interessano poichè dovranno contenere i valori dell'Attack, Decay, Sustain, Release. Nasce immediatamente un primo dubbio: Come mai solo due locazioni per quattro valori?. Il problema è subito risolto: Ogni locazione è adibita a contenere due valori, e più precisamente la locazione 54277 conterrà l'attacco ed il decadimento (Attack/decay) mentre la 54278 sarà riservata al sostenimento e rilascio del suono (Sustain/release); la traduzione non è il massimo ma serve a rendere una prima idea di ciò che tali parametri rappresentano.

In pratica i primi quattro bit sono riservati ad un valore ed i secondi ad un altro.

TABELLA B LE FUNZIONI DEL PROGRAMMA

TASTI	DESCRIZIONE
+	INCREMENTA L'A.D.S.R.
-	DECREMENTA L'A.D.S.R.
->	SELEZIONA IL PARAMETRO
<-	SELEZIONA IL PARAMETRO
F1	SELEZIONA FORMA D'ONDA
F3	SELEZIONA LOCAZIONE MEMO
F4	SELEZIONA LOCAZIONE MEMO
F5	MEMORIZZA A.D.S.R.
F7	LEGGE A.D.S.R.
S	ASCOLTO DEL A.D.S.R.
F	TORNA ALLO STATO MODIF.

tacco più corto ed il valore 15 a quello maggiormente lungo (vedi figura 2).

Al contrario dell'attacco, il decadimento rappresenta il tempo di caduta del suono una volta raggiunto il volume massimo. Tale tempo è condizionato però dal punto di sostenimento selezionato, ovvero la percentuale di volume massimo (quello della locazione 54296) a cui il suono deve essere tenuto.

In pratica se il Sustain è selezionato a 7 (circa una metà del volume massimo) il Decay rapresenterà la velocità secondo la quale il volume si abbasserà dalla soglia massima alla sua metà (figura 3).

Se ne deduce che se la percentuale di volume selezionata corrisponde al cento per cento del volume stesso (attack settato a 15) non ci sarà nessun decadimento anche se quest'ultimo è settato al suo valore massimo (figura 4).

Volendo prendere un riferimento acustico preciso, si può pensare al suono del pianoforte; questo oltre ad essere caratterizzato da un attack abbastanza veloce, ha una percentuale di sostenimento molto bassa, tanto che il suono scende sensibilmente di volume, a differenza del suono di un organo che rimane più o meno allo stesso volume fino a quando non si rilascia il tasto.

TABELLA C CORRISPONDENZA NOTE/TASTI

DO	DO#	RE	RE#	MI	FA	FA#	SOL	SOL#	LA	LA#	SI	DO
Q	2	W	3	E	R	5	T	6	Y	7	U	I

La questione sarebbe presto risolvibile rappresentando i numeri in binario, ma presumo che qualche musicista preferisca avere suggerimenti più pratici piuttosto che una noiosa lezione su tale sistema di numerazione. Infatti esiste una regola precisa che permette di stabilire il valore esatto da inserire senza complessi calcoli. I parametri A.D.S.R. devono essere selezionati in un range (intervallo) 0-15; la regola consiste nell'aggiungere il valore (da 0 a 15) del Decay al valore (da 0 a 15) dell'Attack moltiplicato per 16:

POKE S+5, Decay+Attack*16

Per esempio se vogliamo settare l'attacco col valore 15 e il decadimento con 7 dobbiamo scrivere:

POKE S+5,7+15*16

Lo stesso vale per Sustain/release:

POKE S+6, Release+Sustain*16

L'istruzione seguente setta a 4 il tempo di rilascio ed a 8 il punto di sostenimento:

POKE S+5,4+8*16

L'A.D.S.R. e il volume

Come avete potuto notare si è parlato di tempo di decadimento e punto di sostenimento; cerchiamo di capire cosa significano. Il controllo dell'A.D.S.R, anche se sembra una cosa così complicata, non è niente altro che un controllo di volume più sofisticato; in pratica i parametri ad esso dedicati gestiscono l'andamento del volume nel propagarsi del suono. Attacco, decadimento e rilascio sono parametri di tempo ed il sostenimento rappresenta invece una percentuale del volume generale (vedi figura 1).

Per attacco si intende il tempo che intercorre tra l'innesco della nota ed il raggiungimento del volume massimo settato in precedenza; maggiore è l'attacco, maggiore è l'intervallo di tempo. In pratica un suono come quello di un tamburo è caratterizzato ad esempio da un attacco corto (il tempo per raggiungere il volume massimo è molto breve), mentre il suono di un violino si differenzia per un attacco lungo (il periodo che intercorre tra l'innesco del suono ed il raggiungimento del volume massimo è molto lungo). Se ne ricava che il valore 0 corrisponderà all'at-

stico preciso, si può pensare al suono del pianoforte; questo oltre ad essere caratterizzato da un attack abbastanza veloce, ha una percentuale di sostenimento molto bassa, tanto che il suono scende sensibilmente di volume, a differenza del suono di un organo che rimane più o meno allo stesso volume fino a quando non si rilascia il tasto.

Questo "rilasciare il tasto" corrisponde, tra l'altro, all'avvio della fase di "decay" ovvero di caduta del suono; un suono infatti può essere "chiuso" in modi sensibilmente diversi. Il suono di un tamburo è più secco di quello di una corda di chitarra, il decay di un tamburo è molto veloce, quello della chitarra, al contrario, molto lento (vedi figura 5).

In termini di POKE, occorrerà settare da 0 per un decadimento veloce fino al lentissimo 15.

Anche per il Decay come per il Sustain vale la regola della percentuale di sostegno che annulla. Se il sostenimento è settato a zero, significa, in pratica, che il

decadimento scenderà più o meno velocemente fino alla "chiusura" del suono, eliminando così l'intervento del Decay (figura 6).

Più curve, meno curve

Se volessimo rappresentare graficamente l'evoluzione di un A.D.S.R, indicheremmo con delle curve attacco, decadimento e rilascio e con un punto il Sustain. Saranno inoltre necessari il punto zero (volume nullo) ed il punto 15 rappresentante la soglia massima di volume da raggiungere. Tale punto 15 sarà relazionato al valore settato nella locazione del volume generale, di conseguenza non rappresenterà il massimo assoluto ma un massimo relativo a tale selezione.

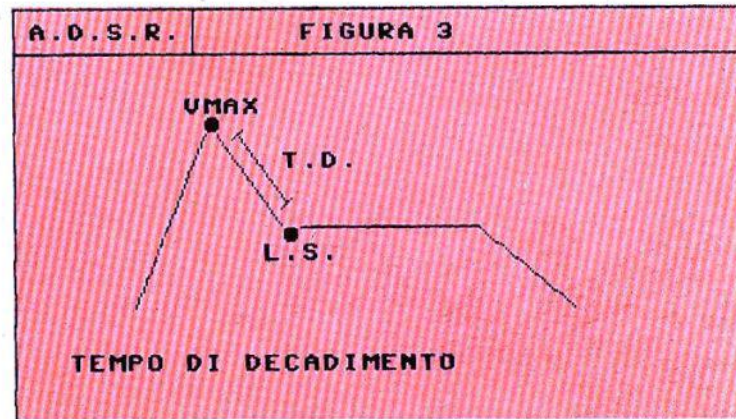
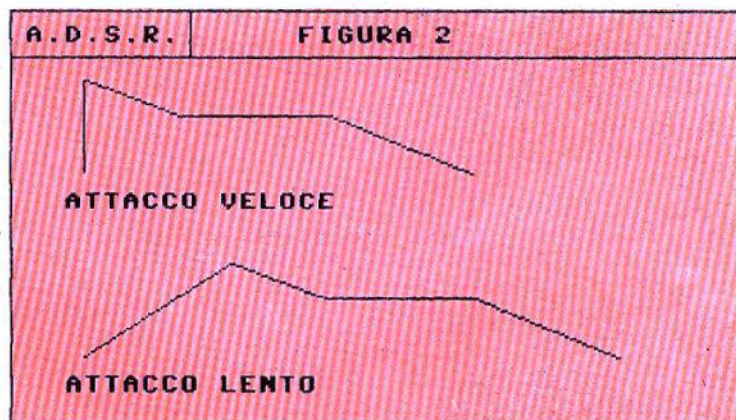
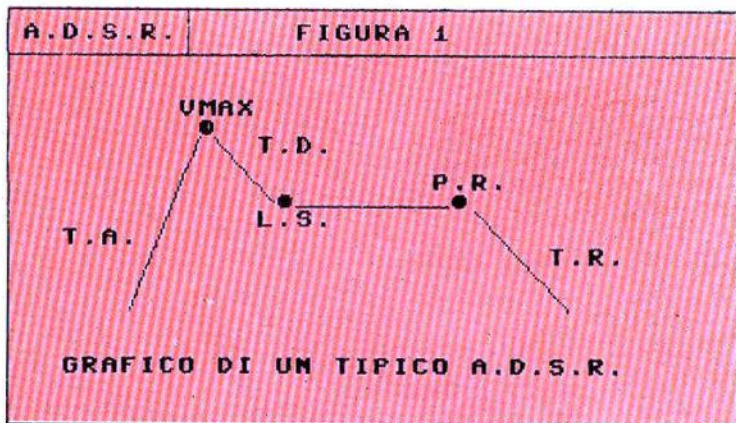
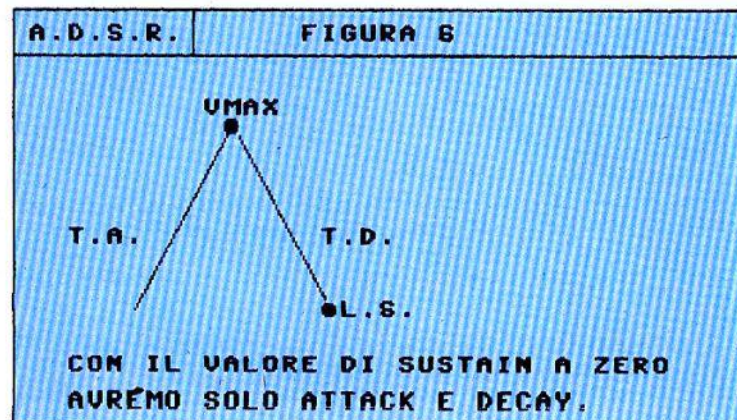
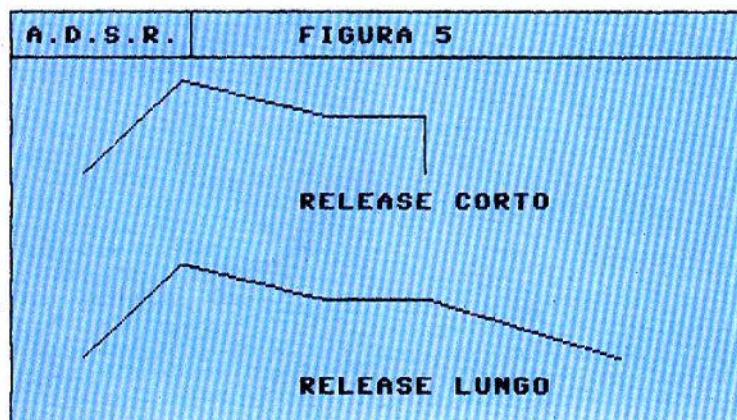
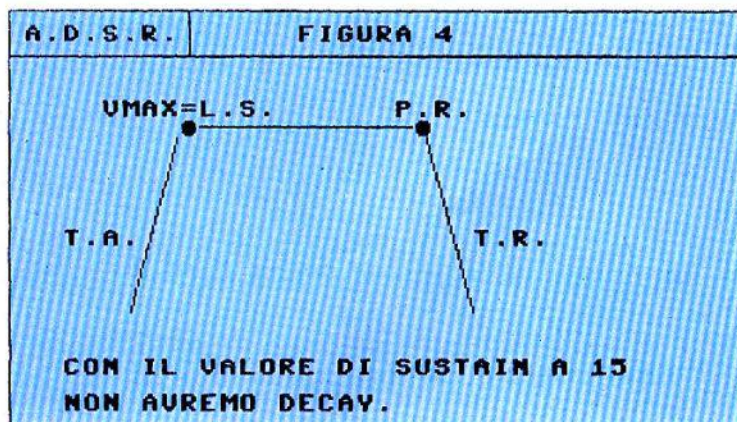
Sull'asse delle Y prenderemo come unità di misura il volume da 0 a 15 e sull'asse delle X i tempi in millisecondi: la curva, ovviamente, rappresenterà l'andamento del suono.

Per questo lavoro ci tornerà utile la tabella B, indicante appunto i tempi di A.D.S.R. Seguendo tale tabella saremo in grado di tracciare grafici rappresentanti i nostri A.D.S.R. senza più rimanere attoniti quando si sente parlare di curva d'attacco o di soglia di sostegno.

Dalla teoria agli esercizi

Il programma di questo articolo ha un duplice uso. Il primo, didattico, ed il secondo di utilità nella creazione di particolari A.D.S.R. Ci siamo preoccupati maggiormente dell'uso didattico, lasciandolo aperto a piccole modifiche atte a farlo diventare un vero e proprio Tool di lavoro.

La sua finalità consiste nel facilitare la modifica dei parametri di Attack/Decay/Sustain/Relaise nonchè di poterne ascoltare i risultati con immediatezza. In poche parole il programma permette di intervenire in tempo reale su tali parametri del suono.

La pratica di questo software permetterà, dunque, di comprendere al meglio l'andamento del volume. Inoltre, e qui risiede la sua maggiore "didatticità", ogni modifica dell'A.D.S.R. verrà rappresentata con un istogramma a barre su video.

Il programma in pratica...

La tabella C contiene una descrizione dei comandi, mentre è possibile avere un'idea della pagina video guardando la figura 7. Il programma avrà dua stati: il primo di intervento ed il secondo di ascolto. Lo stato di intervento, cioè di modifica dei parametri, si imposta automaticamente dando il "RUN". Un cursore a barra inizia a lampeggiare sulla scritta Attack; questo significa che si può intervenire sul parametro di attacco tramite i tasti "+" e "-". Tale intervento incrementerà o decrementerà il valore di Attack sia nella locazione adibita sia nella sua rappresentazione grafica, aggiungendo o togliendo barre al relativo istogramma.

Allo stesso modo si potrà intervenire su Decay, Sustain e Relaise; basterà spostare il cursore con i tasti "CRSR right" e "CRSR left".

Il tasto F1 permetterà di variare la forma d'onda; ad ogni pressione corrisponderà un'oscillazione diversa (triangolare, dente di sega, quadra e rumore). Abbiamo inserito questa variazione per ampliare le possibilità sonore del programma, ma consigliamo di scegliere una singola forma d'onda (escludendo il rumore) per meglio comprendere l'A.D.S.R..

I numeri da 0 a 9 sulla sinistra della pagina video rappresentano dieci locazioni in cui è possibile memorizzare la configurazione di A.D.S.R. selezionata, in modo da poterla riprovare anche dopo molte modifiche. Spostando la freccia con i tasti F3 e shift F3 si sceglierà la locazione in cui registrare i dati, premendo il tasto F5 si memorizzeranno i dati. Una locazione "piena" sarà segnalata con la lettera "M".

Per riprendere i dati registrati, basterà selezionare con la freccia e premere il tasto F7; immediatamente verrà visualizzato il nuovo istogramma e variati i parametri.

Il secondo "stato", quello di ascolto, è selezionabile con la pressione del tasto "S". Potremmo anche chiamarlo stato di esecuzione, poichè sarà possibile "suonare" su una "tastiera" simulata con i tasti del nostro Commodore. Infatti i tasti predisposti a questo lavoro corrisponderanno alle note nella stessa successione in cui si trovano sul pianoforte (Figura 8).

Per finire di suonare e tornare allo stato di modifica, occorrerà premere il tasto "F".

FIGURA 7

...e in teoria

Il listato è diviso in routines; possiamo associare queste a tre gruppi distinti, un gruppo di grafica, uno di elaborazione ed uno musicale.

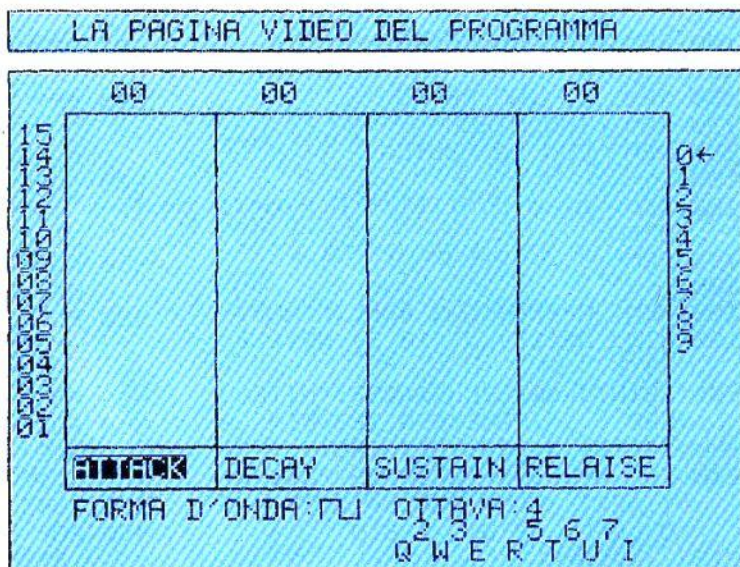
Il gruppo delle routine grafiche contiene:

1/ routine stampa pagina video (380-660);
2/ visualizza una riga di istogramma (840-890);
3/ valori dafault istogrammi (900-940);
4/ scrive valori A.D.S.R. (960-990);
5/ colorazione aree schermo (1000-1060);
6/ stampa forma d'onda (1070-1150);
7/ display locazione memo (1160-1200);
8/ visualizza parametri A.D.S.R. (1260-1320);
9/ cancella istogrammi (1330-1380).

Apponendo opportune modifiche a queste routines si può variare la rappresentazione grafica dell'A.D.S.R. oppure aggiungere la visualizzazione di altri parametri del suono che potrebbero completare la sezione elaborazione e trasformare questo tool didattico in un vero e proprio tool musicale.

Le routines di elaborazione provvedono alla scansione della tastiera ed all'analisi degli inserimenti per procedere alla esecuzione, elaborazione delle modifiche ed alla registrazione delle stesse in opportune locazioni di memoria:

- operazioni preliminari (180-210);
- routine inserimenti (670-830);
- memorizza parametri A.D.S.R. (1210-1250).

Aggiungendo, nella routine inserimenti, i parametri del suono mancanti, quali filtri etc., (quelli delle alchimie, insomma!) e provvedendo alla visualizzazione, questo software potrebbe diventare quasi il massimo. Chiaramente tali modifiche dovranno essere previste anche nelle routines musicali, altrimenti non avrebbero un effetto reale ma solo grafico. Ed a proposito di queste ultime passiamo a proporne la lista:

- lettura delle note (220-270);
- setta i parametri del SID (280-340);
- suona (1390-1530);
- setta A.D.S.R./forma d'onda (1540-1620).

La strada per diventare "alchimisti del suono" è aperta un po' di pratica e presto trasformerete il legno in oro, o meglio il vetro in diamanti, o meglio..... Insomma buon divertimento.

Uso corretto del programma

Come già spiegato nell'articolo, il programma, nonostante debba esser considerato un interessante approccio musicale, potrebbe dare alcune delusioni se usato in modo improprio.

Selezionate a piacimento i valori A.D.S.R e in seguito premete i tasti Q,W,E,R,T,Y,U,I oppure 2,3,5,6,7 in modo da emettere note musicali.

Per esaminare in modo completo l'effetto di una combinazione qualsiasi A.D.S.R è necessario, nel caso di selezione di valori alti, tenere premuto a lungo il tasto.

Ciò perchè bisogna dare il tempo, al circuito elettronico, di "salire" di volume (fase di attack), e di scendere al valore impostato di decay. Solo a questo punto il volume rimane di intensità costante e rimarrà allo stesso livello fino a che non rilascerete il tasto. Da questo momento, infine, entra in azione la fase di rilascio (relaise) e, se non premete altri tasti nel frattempo, il suono diminuirà fino a scomparire.

Quanto detto si verifica, come è intuibile, con valori di attack, decay e relaise molto lunghi. Negli altri casi il fenomeno sonoro può risultare talmente veloce che non farete a tempo a sollevare il dito dal tasto che il suono è già stato emesso.

Per meglio comprendere il funzionamento del SID, provate le seguenti combinazioni e tenete premuto a lungo (oppure no) il tasto relativo alla nota desiderata.

Valori dei registri				Tempo di press.
A.	D.	S.	R.	della nota
15	14	15	15	Molto lungo
15	7	7	15	Molto lungo
10	10	10	10	Lungo
7	7	7	7	Breve
3	3	3	3	Brevissimo

```
100 REM  COMMODORE 64
110 :
120 REM  IL TUO PRIMO
130 REM  INCONTRO
140 REM  CON LA MUSICA
150 :
160 REM  DI M.L.NITTI - D.R.MATT
    URRO
170 REM
180 :
185 REM  OPERAZIONI PRELIMINARI
190 DIM  D$(15),V%(10,5),NO%(13,
    3)
200 PRINTCHR$(5)CHR$(147)"ATTEND
    ERE....":POKE 650,128:REM  A
    BILITA RIPET. DEI TASTI
210 FOR K=55296 TO 56295:POKE K,
    1:NEXT: REM  COLORA DI BIANC
    O LE LOCAZIONI TESTO
215 :
220 REM  LETTURA DELLE NOTE  (TA
    STO,BYTE BASSO,BYTE ALTO)
230 FOR K=1 TO 13:FOR KK=2 TO 3:
    READ NO%(K,KK):NEXTKK,K
240 FOR K=1 TO 13:READ NO%(K,1):
    NEXTK
250 DATA  17,37,18,42,19,63,20,1
    00,21,154,22,227,24,63
260 DATA  25,177,27,56,28,214,30
    ,141,32,94,34,75
270 DATA  62,59,9,8,14,17,16,22,
    19,25,24,30,33
275 :
280 REM  SETTA I PARAMETRI DEL S
    ID
290 S=54272: REM  VALORE DELLA P
    RIMA LOCAZIONE
300 FOR K=S TO S+24:POKE K,0:NEX
    T: REM  AZZERA REGISTRI
310 POKE S+24,15:REM  SETTA IL V
    OLUME AL MASSIMO (VOLUME 0-1
    5)
320 POKE  S+2,0: REM  BYTE BASSO
     PER ONDA QUADRA
330 POKE  S+3,8: REM  BYTE ALTO
     PER ONDA QUADRA
340 POKE  S+4,65: REM  SELEZION
    A FORMA D'ONDA (QUADRA)
350 :
360 REM  ROUTINE STAMPA PAGINA V
    IDEO
380 GOSUB  900:REM  SETTA VALORI
     DI DEFAULT
390 PRINTCHR$(19)+CHR$(144):POKE
     53280,12:POKE 53281,12: RE
    M  COLORI SF./BORDO
400 POKE  1024,112:FOR K=1025 TO
     1025+38:POKE K,64:NEXT:POKE
     1063,110:
410 FOR K=1064 TO 1944 STEP 40:P
    OKE K,93:POKE K+39,93:NEXT
420 POKE  1984,109:FOR K=1985 TO
     1985+38:POKE K,64:NEXT:POKE
     2023,125
430 POKE  1107,112
440 FOR KK=0 TO 3:FOR K=1 TO 7:P
    OKE 1107+K+KK*8,64:NEXT:POKE
     1107+K+KK*8,114:NEXT
450 POKE  1107+K+(KK-1)*8,110
460 FOR KK=1 TO 16:FOR K=1 TO 40
     STEP 8:
470 POKE  1106+KK*40+K,93:NEXTK,
    KK
480 POKE  1747,107
490 FOR KK=0 TO 3:FOR K=1 TO 7:P
    OKE 1747+K+KK*8,64:NEXT:POKE
     1747+K+KK*8,91:NEXT
500 POKE  1747+K+(KK-1)*8,115
510 FOR K=1 TO 40 STEP 8
520 POKE  1786+K,93:NEXTK
530 FOR K=1070 TO 1070+8*3 STEP
    8
540 POKE K,ASC(D$(Y)):POKE K+1,A
    SC(D$(Y)):NEXT
550 POKE  1827,109
560 FOR KK=0 TO 3:FOR K=1 TO 7:P
    OKE 1827+K+KK*8,64:NEXT:POKE
     1827+K+KK*8,113:NEXT
570 POKE  1827+K+(KK-1)*8,125
580 FOR K=0 TO 9:POKE 1220+K*40,
    K+48:NEXT
590 POKE  1221,31
600 PRINTCHR$(19):PRINT:PRINT
610 FOR K=1 TO 15:READ D$(16-K):
    PRINT TAB(1);D$(16-K):NEXT:P
    RINT
620 PRINT TAB(4)"ATTACK"; TAB(12
    )"DECAY"; TAB(20)"SUSTAIN";
    TAB(28);"RELAISE"
630 PRINT:PRINT TAB(4)"FORMA D'O
    NDA:";
635 PRINTCHR$(5)CHR$(18)"..."CHR
```

```
    $(146)CHR$(144)" OTTAVA:4
    "
640 PRINTCHR$(29)"COMANDI:......
    ......"CHR$(18)".2.3...5.6.7
    ..."
650 PRINTCHR$(144)CHR$(29)"S F +
     - F1/3/4/5/7  [RVS]Q W E R
    T Y U I";
660 GOSUB 1070:PRINTCHR$(19):GOS
    UB 1000
670 :
680 REM  ROUTINE INSERIMENTI
700 FOR K=H(C) TO H(C)+6:POKE K,
    PEEK(K)+128:NEXT : REM  LAMP
    EGGIO REVERSE ON
710 FOR K=H(C) TO H(C)+6:POKE K,
    PEEK(K)-128:NEXT : REM  LAMP
    EGGIO REVERSE OFF
720 GET A$:IF A$="" THEN 700
730 IF A$=CHR$(29) AND C<4 THEN
    C=C+1:GOTO 700: REM  CURSORE
     A SINISTRA
740 IF A$=CHR$(157) AND C>1 THEN
     C=C-1:GOTO 700:REM  CURSORE
     A DESTRA
750 IF A$="+" AND Y(C)<15 THEN G
    OSUB 870:GOTO 700: REM  INCR
    EMENTO A.D.S.R.
760 IF A$="-" AND Y(C)>0 THEN GO
    SUB 880:GOTO 700:  REM  DECR
    EMENTO A.D.S.R.
770 IF A$=CHR$(133) THEN GOSUB 1
    070: REM  CAMBIO FORMA D'OND
    A (F1)
780 IF A$=CHR$(134) AND VM<9 THE
    N VM=VM+1:GOSUB 1160:GOTO 70
    0: REM  INCR.MEMO (F3)
790 IF A$=CHR$(138) AND VM>0 THE
    N VM=VM-1:GOSUB 1160:GOTO 70
    0: REM  DECR.MEMO (F4)
800 IF A$=CHR$(135) THEN GOSUB 1
    210:GOTO 700: REM  MEMORIZZA
     A.D.S.R (F5)
810 IF A$=CHR$(136) THEN GOSUB 1
    260:GOTO 700: REM  VISUALIZZ
    A A.D.S.R. (F7)
820 IF A$="S" THEN GOSUB 1390:GO
    TO 700
830 GOTO 700: REM  SUONA
840 :
850 REM  VISUALIZZA UNA RIGA DI
    ISTOGRAMMA
870 Y(C)=Y(C)+1:FOR K=H(C)-40 TO
     H(C)-34:POKE K-Y(C)*40,160:
    NEXT:GOTO 960
880 FOR K=H(C)-40 TO H(C)-34:POK
    E K-Y(C)*40,32:NEXT:Y(C)=Y(C
    )-1:GOTO  960
890 DATA  15,14,13,12,11,10,09,0
    8,07,06,05,04,03,02,01
900 :
910 REM  VALORI DEFAULT ISTOGRAM
    MI
930 I1=1708:I2=I1+8:I3=I2+8:I4=I
    3+8:C=1:D$(0)="00":FO=1:VM=0
    :EM=0
940 H(1)=1788:H(2)=1796:H(3)=180
    4:H(4)=1812:FOR K=1 TO 4:Y(K
    )=0:NEXT:RETURN
950 :
960 REM  SCRIVE VALORI ADSR
980 POKE 1070+(C-1)*8,ASC(LEFT$(
    D$(Y(C)),1))
990 POKE 1071+(C-1)*8,ASC(RIGHT$
    (D$(Y(C)),1)):RETURN
1000 :
1010 REM  COLORAZIONE AREE SCHER
    MO
1030 POKE 56066,0:POKE 56073,0:PO
    KE 56074,0
1040 FOR K=55532 TO 55492+9*40 ST
    EP 40:POKE K,0:NEXT
1050 FOR K=55980 TO 55420 STEP -4
    0:FOR KK=K TO KK+6:POKE KK,0
    :NEXTKK,K
1060 FOR K=55996 TO 55436 STEP -4
    0:FOR KK=K TO KK+6:POKE KK,0
    :NEXTKK,K:RETURN
1070 :
1080 REM  STAMPA FORMA D'ONDA
1100 FOR K=1881 TO 1883:POKE K,32
    :NEXT:FO=FO+1
1110 ON FOGOTO 1130,1140,1150
1120 FO=0:POKE 1881,78:POKE 1882,
    77:RETURN: REM  TRIANGOLARE
1130 POKE 1881,18:POKE 1882,21:PO
    KE 1883,13:RETURN : REM  RUM
    ORE
1140 POKE 1881,79:POKE 1882,76:PO
    KE 1883,116:RETURN :REM  QUA
    DRA
1150 FOR K=1881 TO 1883:POKE K,23
```

```
     3:NEXT:RETURN: REM  DENTE DI
      SEGA
1160 :
1170 REM  DISPLAY LOCAZIONE MEMO
1190 POKE 55492+EM*40,0:POKE 5549
     2+VM*40,1
1200 POKE  1221+EM*40,32:POKE 122
     1+VM*40,31:EM=VM:RETURN
1210 :
1220 REM  MEMORIZZA PARAMETRI ADS
     R
1240 FOR K=1 TO 4:POKE 1180,13:V%
     (VM,K)=Y(K):POKE 1180,141
1250 NEXT:V%(VM,5)=FO:POKE 1180,3
     2:POKE 1222+VM*40,13:RETURN
1260 :
1270 REM  VISUALIZZA PARAMETRI AD
     SR
1290 POKE 1180,150:GOSUB  1330
1300 PS=C:FOR C=1 TO 4:IF V%(VM,C
     )=0 THEN GOSUB 960:GOTO 1320
1310 FOR K1=1 TO V%(VM,C):GOSUB 8
     70:NEXTK1
1320 NEXTC:FO=V%(VM,5)-1:GOSUB 10
     70:C=PS:POKE 1180,32:RETURN
1330 :
1340 REM  CANCELLA ISTOGRAMMI
1360 FOR K=1 TO 4:Y(K)=0:NEXT
1370 FOR K=1708 TO 1708-40*14 STE
     P -40:FOR KK=0 TO 24 STEP 8:
     FOR KH=0 TO 6
1380 POKE K+KK+KH,32:NEXTKH,KK,K:
     RETURN
1390 :
1400 REM  SUONA
1420 NP=0:GOSUB 1540
1430 FOR K=1885 TO 1884+15:POKE K
     ,PEEK(K)+128:NEXT: REM  ACCE
     NDE L'INDICATORE SUONA
1440 IF PEEK(197)=64 THEN 1440: R
     EM  NESSUN TASTO
1450 IF PEEK(197)=21 THEN 1520: T
     AS TO  F PER FINIRE DI SUON
     ARE
1460 NP=PEEK(197):FOR K=1 TO 13:I
     F NP<>NO%(K,1) THEN 1480
1470 POKE S+1,NO%(K,2):POKE S,NO%
     (K,3):GOTO 1490: REM  TROVA
     LA NOTA/TASTO
1480 NEXT:GOTO 1440: REM  NESSUNA
      NOTA/TASTO VALIDA
1490 POKE S+4,W: REM  ACCENDE L'O
     DCILATTORE CON LA FORMA D'ON
     DA SCELTA
1500 IF PEEK(197)=NP THEN 1500: R
     EM  CONTINUA A TENERLA ACCES
     A SE TASTO UGUALE
1510 POKE S+4,W-1:GOTO 1440: REM
      SPEGNE L'OSCILLATORE
1520 POKE S+4,0:REM  SPEGNE L'OSC
     ILLATORE PRIMA DI TORNARE AG
     LI INSERIMENTI
1530 FOR K=1885 TO 1884+15:POKE K
     ,PEEK(K)-128:NEXT:RETURN:REM
       SPEGNE L'INDICATORE S
1540 :
1550 REM  SETTA ADSR/FORMA D'ONDA
1570 POKE S+5,Y(1)*16+Y(2):POKE S
     +6,Y(3)*16+Y(4)
1580 ON FOGOTO 1600,1610,1620
1590 W=17:RETURN: REM  TRIANGOLAR
     E
1600 W=129:RETURN: REM  RUMORE
1610 W=65:RETURN: REM  QUADRA
1620 W=33:RETURN: REM  DENTE DI S
     EGA
```

Enciclopedia di routine

a cura di Alessandro de Simone

Come realizzare l'enciclopedia e utilizzarla nei propri listati.

12100 Protezione software

Questa non è una subroutine "normale", ma un programmino che, una volta usato, diventa inutile.

Va infatti inserito in coda al programma in Basic da proteggere e attivato soltanto quando tale programma è completo. Nel listato principale (da proteggere) va inserita, ogni tre o quattro righe, l'istruzione: REM***

Fatto ciò, si attiverà il protettore con un semplice RUN 12100. Passato un certo tempo (dipendente dalla lunghezza del programma principale), riapparirà il cursore segnalando che la protezione è completata. A questo punto listare il programma sarà molto, ma molto arduo.

Descrizione

12100: imposta un ciclo di ricerca dall'inizio della memoria RAM del Basic (2048 per il C-64 e 4096 per Vic 20) fino alla fine della memoria occupata dal programma, definita dalle locazioni 45 e 46.
12120: cerca nella memoria il token (tcodice) dell'istruzione REM che risulta 147. Se lo trova, passa alla riga successiva altrimenti salta al NEXT della riga 12160.
12140: cerca l'ASCII dei tre asterischi (=42). In caso positivo, esegue la POKE alla fine della riga, inserendo al posto dell'ultimo asterisco il carattere 19 (=Home).

Pietromassimo Pasqui

```
100 REM  ESEMPIO D'USO
105 REM  PROTEZIONE SOFTWARE
110 :
120 PRINT"QUESTE SONO SOLTANTO"
130 REM ***
140 PRINT"RIGHE DI PROVA"
150 REM ***
160 PRINT"DEL PROGRAMMA"
170 REM ***
180 GOSUB 12100
999 END
9999 :
12100 FOR X1=2048 TO PEEK(46)*256+
      PEEK(45)
12105 REM  PER VIC 20: FORX1=4096T
      OPEEK(46)*256+PEEK(45)
12120 IF PEEK(X1)<>143 THEN 12160
12140 IF PEEK(X1+1)=42 AND PEEK(X1
      +2)=42 AND PEEK(X1+3)=42 THE
      N POKE X1+3,19
12160 NEXT:END
12190 REM  VARIABILI:X1=CICLO RICE
      RCA
12191 REM  INSERIRE NEL PROGRAMMA
12192 REM  PRINCIPALE OGNI 3 - 4
12193 REM  RIGHE L'ISTRUZIONE REM*
      **
12195 REM  AL FINE DI ATTIVARE
12196 REM  IL PROTETTORE CON: RUN
      12100
12199 REM  PROTETTORE BASIC
```

12200 Numeri congrui

Questa subroutine serve per verificare se due numeri sono congrui rispetto a un numero assegnato (modulo). Se, ad esempio, due valori ("A" e "B") sono due angoli e si vuol verificare se sono tra loro congrui di modulo 360, si digiteranno i due numeri e il modulo ottenendo la risposta cercata.

Esempi

Con modulo 360 i numeri 52 e 412 sono congrui perchè 360+52=412.

Modulo 360: 31 e 1111 sono congrui perchè: 360*3+31=1111.

Modulo 17: 18 e 103 sono congrui perchè: 17*5+18=103.

Una volta affidati alla subroutine i due numeri con le variabili X1, X2 e il modulo con la variabile X5, la risposta sarà data dal contenuto della variabile X4:

X4=0 falso (valori non congrui)
X4=1 vero (numeri congrui)

Pietromassimo Pasqui

```
100 REM  ESEMPIO D'USO
102 REM  NUMERI CONGRUI
110 INPUT "MODULO=";X5:INPUT "PR
    IMO N.=";X1:INPUT "SECONDO N
    .=";X2
120 GOSUB 12200:IF X0$="ERR" THE
    N PRINT"ERRORE":GOTO 110
130 IF X4=1 THEN PRINT"NUMERI CO
    NGRUI"
140 IF X4=0 THEN PRINT"NUMERI NO
    N CONGRUI"
```

A i lettori che hanno acquistato per la prima volta questo numero di Commodore Computer Club, illustriamo qui di seguito, in breve, i vantaggi derivanti dalla raccolta proposta. Questa, a pensarci bene, è la versione "superiore" della rubrica "1 RIGA" e potrebbe anche denominarsi... "Una schermata"!

Oltre che utili per costituire un'enciclopedia, i brevissimi sottoprogrammi pubblicati su ogni numero, sono anche validissimi strumenti di studio per coloro che desiderano approfondire le proprie conoscenze del Basic, esaminando, senza fatica, particolari routine o insolite tecniche di programmazione.

- Dato che può esser "chiamata" più di una volta nel corso di un programma, nessuna routine contiene istruzioni del tipo DATA oppure DIM, allo scopo di non creare confusione col listato principale.
- Nessuna routine può far riferimento ad altre routine dell'enciclopedia.
- Nessuna routine può contenere variabili "banali" (A, A$, eccetera), ma solo variabili poco usate (X1$, X8, Y0%, eccetera).
- Ogni routine deve apparire, **per intero,** sullo schermo del computer e consentire, proprio per questo motivo, di essere esaminata comodamente.
- Ogni routine deve esser numerata secondo uno standard che ha la particolarità di esser ricordato facilmente:

Righe	Contenuto
XXY00	Prima riga del sottoprogramma
XXY89	Ultima riga utile del sottoprogramma
XXY90 REM	Prima riga di spiegazioni
XXY99 REM	Nome della subroutine

in cui XX sono due valori variabili da 10 a 63; Y è un carattere numerico compreso tra 0 e 9.

Qualsiasi subroutine, in altre parole, inizia con un numero, di cinque caratteri, che termina **sempre** con "00". La stessa subroutine, d'altra parte, ha l'ultima riga numerata con "99". Digitando, ad esempio: LIST 10800-10899
si avrà la certezza di veder apparire sullo schermo, **per intero,** la routine il cui nome si trova nella riga 10899.

Prima di accedere alla routine, è necessario assegnare, alle variabili indicate con REM da riga XXY89 a XXY98, particolari valori per il suo corretto funzionamento. Al "ritorno" una o più variabili conterranno il risultato dell'elaborazione.

In questo modo, per esser più chiari, è possibile simulare alcuni comandi di versioni Basic avanzate oppure, addirittura, creare nuove e inedite istruzioni. Ad esempio, il comando: SOUND 1,800,500
che, nel C-16, riproduce un suono di tonalità 800 tramite la voce 1 per la durata 500, potrebbe venir riprodotta, in un'ipotetica subroutine per il Commodore 64, con: X1=1:X2=800:X3=500:GOSUB12400
nell'ipotesi, ovviamente, che la routine in oggetto sia allocata da riga 12400 a 12499.

I listati pubblicati "girano" su ogni computer, salvo dove indicato diversamente.

E' ovvio che nel caso del Vic-20, (che, come è noto, ha uno schermo di soli 506 caratteri), le subroutine "universali" funzionano correttamente, ma non possono apparire per intero in una sola schermata.

Per quanto riguarda la digitazione, si tenga presente che sulla rivista, per motivi di chiarezza, i comandi e le istruzioni Basic sono separati tra loro da spazi bianchi. Nel digitare le linee di programma, pertanto, è opportuno ignorarli altrimenti si rischia di non restare in una sola schermata. Se, per esempio, leggete:

12100 X1=34: X2 = SQR(X3) + LOG(X1)

digitate nel modo seguente:

12100 X1=34:X2=SQR(X3)+LOG(X1)

senza, cioè, alcun carattere di separazione tra comandi ed istruzioni.

Collaborazione dei lettori

La collaborazione dei lettori è gradita, purchè si provveda a inviare **almeno** tre sottoprogrammi per volta, su nastro, disco oppure output di stampante. I listati di routine che non rispettano lo standard adottato non potranno esser presi in considerazione.

Tutti i lavori pubblicati verranno compensati con prodotti della Systems Editoriale (cassette di programmi, libri, abbonamenti, copie arretrate, eccetera).

```
  150 GOTO 110
  999 :
 9999 END
12200 X1=ABS(X1):X2=ABS(X2)
12203 X0$="":IF X5=0 THEN X0$="ERR
      ":RETURN
12205 IF X1=<X5 AND X2=<X5 THEN 12
      230
12210 IF X1=<X5 AND X2>X5 THEN 122
      45
12215 IF X1>X5 AND X2=<X5 THEN 122
      55
12220 IF X1>X5 AND X2>X5 THEN 1226
      5
12225 END
12230 X4=0:IF X1=X2 THEN X4=1
12235 IF (X1=0 AND X2=X5) OR (X1=X
      5 AND X2=0) THEN X4=1
12240 RETURN
12245 X3=INT(X2/X5):X3=X3*X5:X2=X2
      -X3
12250 GOTO 12230
12255 X3=INT(X1/X5):X3=X3*X5:X1=X1
      -X3
12260 GOTO 12230
12265 X3=INT(X1/X5):X3=X3*X5:X1=X1
      -X3
12270 X3=INT(X2/X5):X3=X3*X5:X2=X2
      -X3
12275 GOTO 12230
12290 REM  X1,X2=NUMERI INSERITI
12291 REM  X5=MODULO <>0
12292 REM  X3=VARIABILE PER CALCOL
      O
12293 REM  X4=1 VERO; X4=0 FALSO
12299 REM  VERIFICA NUMERI CONGRUI
```

50000 Legge blocchi liberi

Ricordate (C.C.C. 24) la routine che consentiva di leggere il numero di blocchi liberi presenti su un dischetto? Un lettore ne propone una versione decisamente migliore e più veloce che, come è ovvio, suggeriamo caldamente illustrandone i principi di funzionamento.

50000: la locazione X4 conta quanti file sono finora stati aperti. Da X5+1 in poi sono memorizzati i numeri con i quali i file sono aperti.
50010-50030: si cerca un numero X2 tale che il comando OPEN X2 (eccetera) non provochi il messaggio FILE OPEN ERROR.
50010: se non ci sono file aperti, si impone X2=1.
50020: altrimenti si aprono due cicli. Il ciclo in X3 non può terminare col NEXT di questa riga; tale NEXT chiude, comunque, anche il ciclo in X2.
50030: quando il ciclo X2 si chiude in questa linea, il valore di X3 è un numero di file lecito e viene inserito in X2. Non serve chiudere il ciclo in X3, che sarà chiuso dal RETURN.
50040: l'apertura di un qualunque file su disco provoca l'aggiornamento automatico delle locazioni "importanti" nella RAM del 1541 (questa operazione è necessaria: il drive non può sapere se, ad esempio, abbiamo cambiato dichetto!). Il file è quindi subito richiuso...
50050: ...e con lo stesso numero viene aperto il canale 15, sul quale il comando Memory Read (M-R) legge tre locazioni del 1541 a partire dalla $02FA.
50060: i contenuti della prima e della terza danno rispettivamente la parte bassa e la parte alta del numero di blocchi liberi.
50070: il canale 15 non deve essere chiuso. Con CLOSE X2 si otterrebbe, infatti, la chiusura automatica di tutti i file finora aperti sul disco. Per non lasciare aperto il file X2 basta però diminuire di uno il numero di file che il computer ritiene siano aperti e il gioco è fatto.

Il programma non ha un test che garantisca contro il "TOO MANY FILES", ma i lettori più attenti possono facilmente aggirare l'ostacolo.

Claudio Baiocchi

```
 100 REM  ESEMPIO D'USO
 110 REM  LEGGE BLOCKS FREE
 115 REM  SU DRIVE 1541
 120 :
 130 GOSUB 50000:PRINTX1"BLOCCHI
     LIBERI":PRINT664-X1"BLOCCHI
     OCCUPATI"
 999 :
9999 END
```

```
50000 X4=152:X5=600:REM  X4=151 E
      X5=1288 SU C-16 E PLUS-4
50010 X1=PEEK(X4):IF X1=0 THEN X2=
      1:GOTO 50040
50020 FOR X3=1 TO 255:FOR X2=1 TO
      X1:IF PEEK(X5+X2)=X3 THEN NE
      XTX3
50030 NEXTX2:X2=X3
50040 OPEN X2,8,0,"$":CLOSE X2
50050 OPEN X2,8,15:PRINT#X2,"M-R"C
      HR$(250)CHR$(2)CHR$(3)
50060 GET #X2,X1$,X2$,X3$:X1=ASC(X
      3$)*256+ASC(X1$)
50070 POKE X4,PEEK(X4)-1:RETURN
50090 REM  VARIABILI:X1,X1$,X2,X2$
      ,X3,X3$,X4,X5;X1=N.BLOCKS FR
      EE
50091 REM  SERVE POTER APRIRE ALME
      NO UN FILE SU DISCO
50099 REM  LEGGE N. BLOCKS FREE
```

50200-50300 File relativi

Questa puntata di Enciclopedia di routine è dedicata in gran parte a utility per la gestione dei file relative: creazione, scrittura e lettura dei record. Speriamo che ciò contriubuisca a eliminare l'alone di mistero che circonda l'argomento e vi permetta finalmente di utilizzare una tecnica di memorizzazione dati, fra le più efficaci oggi disponibili.

Non tutti sanno che con un computer Commodore è possibile creare diversi tipi di file. Ogni tanto se ne parla, ma molto spesso in maniera così nebulosa che, anzichè portare qualche chiarimento, non si fa altro che generare ulteriore confusione.

I diversi tipi di file sono solitamente contraddistinti dalle sigle:

PRG: programma. I soliti e familiari programmi.
SEQ: sequenziale. Possiamo registrare o leggere dati solo in maniera sequenziale. Per analogia li paragoniamo a un libro senza indice: se per ipotesi dobbiamo cercare un argomento che ci interessa, bisogna cominciare dall'inizio e sfogliarne le pagine, fino a trovare quella desiderata.

Veramente molto scomodo! Senza contare la possibilità di arrivare all'ultima pagina e non aver trovato ombra di quanto cercato.

Nonostante ciò, i file sequenziali sono molto usati e spesso da preferirsi, per la loro semplicità e immediatezza.
USR: user. Sono normalissimi file sequenziali. La sigla USR sta solo a distinguere o mettere in evidenza un determinato file.

REL: relative. Ritorniamo al nostro libro. Dotiamolo di un indice. Consideriamo inoltre la possibilità di aggiungere o togliere pagine, anche in posizioni intermedie: ecco i file relativi.

Si intuisce subito che questi ultimi sono molto più versatili e potenti. Conoscendo il numero della pagina (l'indirizzo del record), andiamo a cercare a colpo sicuro ciò che interessa. Un'altra cosa abbastanza intuitiva: non possono essere utilizzati con un registratore, ma solo con un lettore di dischi. Questo proprio per conformazione fisica delle cassette: pensate alla differenza tra una cassetta musicale e un disco 33 giri...

Flavio Molinari

50200 Crea file relative

(Per qualsiasi Commodore e drive 1541)

Il primo passo da compiere per creare un qualsiasi archivio, è definirne fisicamente lo spazio. Sui dischetti le cose funzionano alla stessa maniera: prima di effettuare qualsiasi operazione, di lettura o scrittura, dobbiamo creare un file composto tutto da record vuoti.

Seguiamo l'esempio d'uso.

NOME DEL FILE ?

Un nome a scelta, purchè non sia una stringa nulla.

NUMERO RECORD ?

Da non confondere con i record fisici del dischetto. La classica agenda elettronica con, ad esempio, 100 nominativi, conterrà come minimo 100 record.

LUNGHEZZA RECORD ?

Numero di caratteri per ogni record. Può valere come massimo 255 (compreso l'ultimo carattere separatore). Attenzione però a non superare la capacità del dischetto. E' bene fare prima un po' di conti:

● il valore NUMERO RECORD * LUNGHEZZA non deve superare il numero di blocchi liberi. Ce ne assicuriamo utilizzando una routine (numerata da 50000 a 50099) pubblicata proprio in questa rubrica (n. 24 C.C.C.: Legge blocchi liberi).

Il nostro lavoro finisce qui. Ora tocca al computer fare il resto:

● prepara la stringa che andrà in scrittura (X2$), con degli asterischi (un qualsiasi altro carattere andrà bene);

● OPEN 15,8,15,"I"

inizializza. E' consigliabile farlo ogni volta che si accede al drive;

- OPEN 2,8,2,X1$+",L,"+CHR$(X4)
apre il canale n. 2 per il file X1$;
- ",L," indica il tipo di file: relativo;
- CHR$(X4)=codice ASCII della lunghezza del record;
- PRINT # 15,"P"+CHR$(2)
+CHR$(X8)+CHR$(X7)CHR$(1)
posiziona il puntatore "P" al canale 2, al record n. X7*256+X8 e al byte n.1;
- PRINT # 2,X2$
finalmente scrive la stringa X2$.

Per tranquillità, leggiamo il canale d'errore:

INPUT 15,Y1,Y1$

Se Y1=0 (oppure, solo per i relativi, Y1=50), tutto è OK!

Flavio Molinari

```
  100 REM  ESEMPIO D'USO
  110 REM  CREA FILE RELATIVO
  115 REM  QUALSIASI COMMODORE
  120 REM  E DRIVE 1541
  130 :
  140 INPUT "NOME FILE";X1$
  150 IF X1$="" OR LEN(X1$)>16 THE
      N 140:REM  NOME FILE NON VAL
      IDO
  160 INPUT "NUMERO RECORD";X2
  170 INPUT "LUNGHEZZA RECORD";X3
  180 IF X3>254 THEN 170:REM  MAX
      LUNGHEZZA RECORD=254 CARATTE
      RI
  190 X2$="":FOR XX=1 TO X3-1
  200 X2$=X2$+"*":NEXT
  210 GOSUB 50000:REM  LEGGE BLOCC
      HI LIBERI
  220 IF X3*X2/256>X9 THEN PRINT"F
      ILE TROPPO LUNGO":END
  230 GOSUB 50200
  240 IF X0$<>"" THEN PRINTY1;Y1$:
      END
  250 PRINT"O.K. FILE "X1$" CREATO
      "
  260 END
  270 :
 9999 END
50000 CLOSE 1:OPEN 1,8,0,"$"
50010 GET #1,Y1$,Y2$
50020 GET #1,Y1$,Y2$
50025 GET #1,Y1$,Y2$
50027 X9=0:IF Y1$<>"" THEN X9=ASC(
      Y1$)
50028 IF Y2$<>"" THEN X9=X9+ASC(Y2
      $)*256
50030 GET #1,Y2$:IF ST THEN 50050
50035 IF Y2$<>CHR$(34) THEN 50030
50040 GET #1,Y2$:IF Y2$<>CHR$(34)
      THEN 50040
50045 GET #1,Y2$:IF Y2$=CHR$(32) T
      HEN 50045
50047 GET #1,Y2$:IF Y2$<>"" THEN 5
      0047
50049 IF ST=0 THEN 50020
50050 CLOSE 1:RETURN
50053 :
50054 :
50200 OPEN 15,8,15,"I":REM  INIZIA
      LIZZA
50205 X4=X3+1:OPEN 2,8,2,X1$+",L,"
      +CHR$(X4):GOSUB 50255
50210 IF X0$<>"" THEN 50250
50215 FOR XX=X2 TO 1 STEP -1
50220 X7=INT(XX/256):X8=XX-X7*256
50225 PRINT#15,"P"CHR$(2)+CHR$(X8)
      +CHR$(X7)+CHR$(1):GOSUB 5025
      5
50230 IF X0$<>"" THEN 50250
50235 PRINT#2,X2$:GOSUB 50255
50240 IF X0$<>"" THEN 50250
50245 NEXT:CLOSE 2:GOSUB 50255
50250 CLOSE 2:CLOSE 15:RETURN
50255 X0$="":INPUT#15,Y1,Y1$:IF Y1
      =0 OR Y1=50 THEN RETURN
50260 X0$="ERR":RETURN
50290 REM  X4,X3,X1$,X0$,XX,X2,X7,
      X8,X2$
50291 REM  X1$: NOME FILE RELATIVO
50292 REM  X2: NUMERO RECORD
50293 REM  X3: LUNGHEZZA RECORD
50294 REM  X2$: RECORD IN SCRITTUR
      A
50295 REM  Y1,Y1$: MESSAGGI CANALE
       ERR
50299 REM  CREA FILE RELATIVE
```

50300 Scrive su file relative

Per qualsiasi Commodore e drive 1541

Dopo aver creato un file (composto da record "vuoti"), il passo seguente da compiere è quello di... scriverci qualcosa. Per semplicità, in questa sede consideriamo i record costi-

tuiti da un solo campo: l' eventuale suddivisione sarà compito del programmatore che intendesse farne uso. Prendendo il caso della solita agenda elettronica, la stringa che andrà in scrittura (X2$, nell'esempio d'uso), sarà così composta:

NM$="Rossi Mario"
IN$="Viale dei Mille 1"
CI$="Milano"
TL$="1234567"

X2$=NM$+IN$+CI$+TL$

Per un totale di 25+30+12+12+79 caratteri.
Andiamo oltre:

NOME DEL FILE?
nome di un file presente sul disco.
LUNGHEZZA RECORD?
lunghezza assegnata precedentemente in fase di creazione.
SCRIVI SU RECORD N.?
deve essere compreso tra 1 e il numero max di record disponibili.
SCRIVI STRINGA?
vedere sopra quanto detto sulla variabile X2$.

Senza pretendere di ottenere le stesse prestazioni di un computer professionale, con questa routine vogliamo per lo meno avvicinarci come comodità d'uso. In un linguaggio Basic più evoluto di quello disponibile su C-64, la procedura per scrivere su un file sarebbe:

APRI nomefile, lugh. record xxx
SCRIVI record yyy,"stringa"

Tutte le altre operazioni, come: puntatori da posizionare, calcolo dei byte bassi e alti del record eccetera, sono tipici lavori da computer. Non è il caso di angosciare il programmatore con simili problemi!

Flavio Molinari

```
100 REM  ESEMPIO D'USO
110 REM  SCRIVE SU FILE RELATIVE
120 REM  PER UN CORRETTO FUNZION
    AMENTO
130 REM  ASSEGNARE ALLE VARIABIL
    I:
140 REM  X1$= NOME FILE
150 REM  X3= LUNGUEZZA RECORD
160 REM  GLI STESSI VALORI DI UN
     FILE
170 REM  PRECEDENTEMENTE CREATO
180 :
190 FOR XX=1 TO 254:X9$=X9$+CHR$
    (32):NEXT
220 INPUT "NOME DEL FILE";X1$
240 IF X1$="" OR LEN(X1$)>16 THE
    N 220
260 INPUT "LUNGEZZA RECORD";X3
270 INPUT "SCRIVI SUL RECORD N."
    ;X2:IF X2<1 THEN 270
280 INPUT "SCRIVI STRINGA";X2$:I
    F X2$="" THEN 280
300 X2$=LEFT$(X2$+LEFT$(X9$,254-
    LEN(X2$)),X3)
320 GOSUB 50300
340 IF X0$<>"" THEN PRINTY1;Y1$:
    END
360 PRINT"O.K.":GOTO 270
370 :
9999 END
50300 X4=X3+1:OPEN 15,8,15,"I":X0$
      =""
50305 OPEN 2,8,2,X1$:GOSUB 50345:C
      LOSE 2
50310 IF X0$<>"" THEN CLOSE 2:CLOS
      E 15:RETURN
50315 OPEN 2,8,2,X1$+",L,"+CHR$(X4
      )
50320 X7=INT(X2/256):X8=X2-X7*256
50325 PRINT#15,"P"+CHR$(2)+CHR$(X8
      )+CHR$(X7)+CHR$(1)
50330 PRINT#2,X2$
50335 PRINT#15,"P"+CHR$(2)+CHR$(X8
      )+CHR$(X7)+CHR$(1)
50340 GOSUB 50345:CLOSE 2:CLOSE 15
      :RETURN
50345 INPUT#15,Y1,Y1$:IF Y1=0 OR Y
      1=50 THEN RETURN
50350 X0$="ERR":RETURN
50390 REM  VARIABILI: X1$,X2,X3,X2
      $,Y1,Y1$,X7,X8,X4,X0$
50392 REM  X1$: NOME FILE
50394 REM  X2: N. RECORD DA SCRIVE
      RE
50396 REM  X3: LUNGHEZZA RECORD
50398 REM  X2$: STRINGA DA SCRIVER
      E
50399 REM  SCRIVE STRINGA SU FILE
      RELATIVE
```

50400 Legge file relative

Per qualsiasi Commodore e drive 1541

Le stesse considerazioni viste sopra valgono anche per la lettura dei record.

Alle richieste:

NOME DEL FILE?
LUNGHEZZA DEI RECORD?
assegnare i valori dati in fase di creazione.

N. RECORD DA LEGGERE?
il contenuto del record viene letto (GET #) e presentato sul video.

A questo punto non è possibile fare una routine universale che valga per tutti i casi. Sarà compito di chi ha creato l'archivio decidere se è meglio usare l'istruzione GET # o INPUT #, separare i campi tra loro e, in generale, pensare a tutti quei problemi che non possono essere risolti con delle semplici routine, a partire dalla fase di studio che deve sempre precedere un problema spesso complesso come la gestione degli archivi.

Flavio Molinari

```
  100 REM  ESEMPIO D'USO
  110 REM  LEGGE DA FILE RELATIVE
  120 REM  PER UN CORRETTO FUNZION
      AMENTO
  130 REM  ASSEGNARE ALLE VARIABIL
      I:
  140 REM  X1$= NOME FILE
  150 REM  X3= LUNGHEZZA RECORD
  160 REM  GLI STESSI VALORI DI UN
       FILE
  170 REM  PRECEDENTEMENTE CREATO
  180 :
  200 INPUT "NOME FILE";X1$
  210 IF X1$="" THEN 200
  220 INPUT "LUNGHEZZA DEI RECORD"
      ;X3
  240 INPUT "N. RECORD DA LEGGERE"
      ;X2:IF X2<1 THEN 240
  260 GOSUB 50400
  280 IF X0$<>"" THEN PRINTY1;Y1$:
      END
  300 PRINTX2$:GOTO 240
 9999 :
50400 X4=X3+1:OPEN 15,8,15,"I":X0$
      =""
50405 OPEN 2,8,2,X1$:GOSUB 50445:C
      LOSE 2
50410 IF X0$<>"" THEN CLOSE 15:RET
      URN
50415 OPEN 2,8,2,X1$+",L,"+CHR$(X4
      )
50420 X7=INT(X2/256):X8=X2-X7*256
50425 PRINT#15,"P"+CHR$(2)+CHR$(X8
      )+CHR$(X7)+CHR$(1)
50427 GOSUB 50445:IF X0$<>"" THEN
      CLOSE 2:CLOSE 15:RETURN
50430 X2$="":FOR XX=1 TO X3:GET #2
      ,XX$:X2$=X2$+XX$:NEXT
50435 CLOSE 2:CLOSE 15:RETURN
50440 GOSUB 50445:CLOSE 2:CLOSE 15
      :RETURN
50445 INPUT#15,Y1,Y1$:IF Y1=0 OR Y
      1=50 THEN RETURN
50450 X0$="ERR":RETURN
50490 REM  VARIABILI: X1$,X2,X3,Y1
      ,Y1$,X4,X7,X8,X0$,XX$,X2$
50492 REM  X1$: NOME FILE
50494 REM  X2$: RECORD IN LETTURA
50496 REM  X3: LUNGHEZZA RECORD
50499 REM  LEGGE DA FILE RELATIVE
```

50500 Visualizza file

Per qualsiasi Commodore e drive 1541

Avete un file su dischetto, ma avete dimenticato cosa contiene? Un testo scritto con word-processor, ma non sapete quale?

Questa routine fa al caso vostro. Date il nome del file (X1$) e tenete premuta la barra spaziatrice, per visualizzarne il contenuto. Il tasto Return consente di finire in qualsiasi momento.

Può essere visualizzato qualsiasi tipo di file, naturalmente si prestano meglio quelli di tipo testo. Nei programmi (in Basic, non in linguaggio macchina!), saranno comprensibili solo i nomi delle variabili, le stringhe alfanumeriche e i caratteri dopo le REM. Gli eventuali codici che non possono essere rappresentati sullo schermo, vengono visualizzati con un asterisco in campo inverso.

Flavio Molinari

```
100 REM  ESEMPIO D'USO
110 REM  LEGGI CONTENUTO DI
120 REM  FILE SEQ, PRG, USR E RE
    L
130 REM  QUALSIASI COMPUTER E DR
    IVE 1541
```

```
  140 :
  150 INPUT "NOME FILE DA LEGGERE"
      ;X1$
  160 PRINT"PREMI SPAZIO PER CONTI
      NUARE LA VISUALIZZAZIONE ";
  170 PRINT"OPPURE <RETURN> PER FI
      NIRE":PRINT
  180 GOSUB 50500
  190 IF X0$<>"" THEN PRINTY1;Y1$
  200 :
 9999 END
50500 OPEN 15,8,15:OPEN 2,8,2,X1$
50505 X2$=CHR$(18)+CHR$(42)+CHR$(1
      46)
50510 GOSUB 50540:IF X0$<>"" THEN
      CLOSE 2:CLOSE 15:RETURN
50515 GET XX$:IF XX$=CHR$(13) THEN
       CLOSE 2:CLOSE 15:RETURN
50520 IF XX$<>CHR$(32) THEN 50515
50525 GET #2,X1$
50530 IF (X1$<CHR$(32) OR X1$>CHR$
      (127)) AND (X1$<CHR$(160)) T
      HEN X1$=X2$
50535 PRINTX1$;:GOTO 50515
50540 INPUT#15,Y1,Y1$
50545 IF Y1=0 THEN RETURN
50550 X0$="ERR":RETURN
50590 REM  VARIABILI: X1$,X2$,X0$,
      XX$,Y1,Y1$
50592 REM  X1$: NOME FILE
50594 REM  VISUALIZZA UN ASTERISCO
       IN
50596 REM  CAMPO INVERSO SE CARATT
      ERE
50598 REM  NON VISUALIZZABILE
50599 REM  NOME: VISUALIZZA FILE
```

Elenco delle routine pubblicate

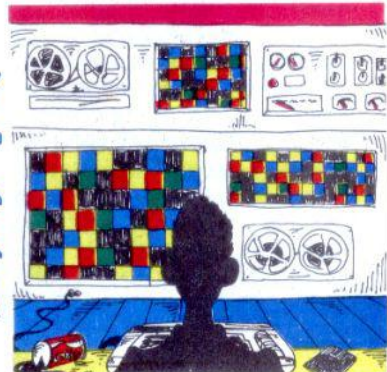
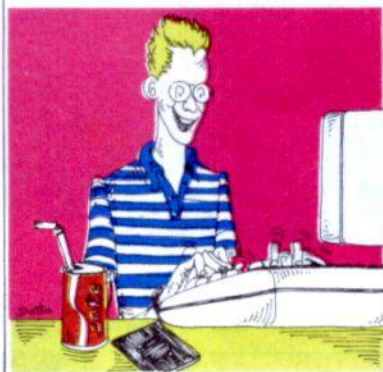
Qui di seguito viene riportato l'indice di tutte le routine finora pubblicate. Fra parentesi il numero di Commodore Computer Club su cui sono uscite.

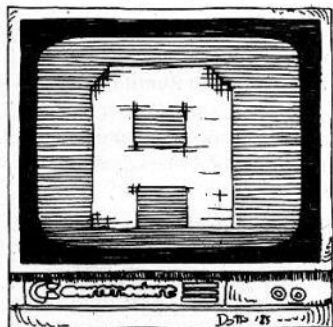
per i PROGETTI SPECIALI della Systems Editoriale

collaboratori a tempo pieno oppure parziale.

La persona ideale:

- **risiede a Milano o nel suo hinterland;**
- **è REALMENTE esperta in Linguaggio Macchina e Assembler;**
- **è in grado di sviluppare autonomamente programmi di qualsiasi tipo;**
- **è in possesso di un sistema completo (computer Commodore, drive, registratore, ecc.);**
- **è interessato ad acquisire esperienza anche su altri sistemi (MSX, Sinclair, MS/DOS, ecc.)**

Per un primo contatto TELEFONARE alla Systems Editoriale (tel. 8467348) il martedì o venerdì pomeriggio (dopo le 16) chiedendo dell'ingegner de Simone.

Un nuovo look per i caratteri del C 16

di Diego e Federico Canetta

Il C-16 conta ormai parecchie migliaia di utenti. Le potenzialità del piccolo computer sono pienamente riconosciute, ma, a dispetto delle numerose possibilità offerte dalla macchina, manca ancora una valida letteratura che ne descriva efficacemente le caratteristiche e il funzionamento.

In questo articolo vogliamo illustrare un argomento basilare per un buon sfruttamento del computer, cioè la possibilità di programmare i caratteri.

Quanto scritto riguarda tecniche di programmazione avanzata ed è quindi adatto a chi ha già familiarità con la programmazione Basic.

Chi, tuttavia, non è in grado di comprendere l'articolo, può limitarsi a digitare il listato del gioco e divertirsi senza tanti problemi.

I caratteri presenti nella ROM

In questo paragrafo spieghiamo brevemente come il computer faccia apparire i caratteri sullo schermo.

Impariamo a realizzare caratteri personalizzati per le applicazioni più disparate.

A partire dalla locazione di memoria 53444 è presente una zona di memoria ROM (quindi di sola lettura) in cui sono registrati i caratteri preprogrammati del C-16. Quando il calcolatore viene acceso, si viene a trovare nel cosiddetto modo **carattere standard,** per cui quando premiamo, ad esempio, il tasto "A", il microprocessore del C-16 preleva il carattere "A" dalla zona di memoria citata (sotto forma di numeri) e lo porta sullo schermo. In tal modo vedremo apparire la lettera A sul video.

Attraverso opportuni artifizi, si può far leggere al microprocessore una zona di memoria diversa dalla ROM, ad esempio, una zona di RAM in cui abbiamo avuto l'accortezza di depositare caratteri personalizzati, ottenendone così la visualizzazione.

Prima però di parlare della scrittura dei caratteri, occorre precisare meglio il modo in cui essi sono formati.

I caratteri del C-16

Se osserviamo attentamente i caratteri che appaiono sullo schermo, notiamo che sono formati da una griglia di 8x8 puntini, chiamati "dot". Ogni punto della griglia può essere acceso (ON) o spento (OFF). Per chiarezza, nelle figure sono state numerate tutte le colonne con numeri progressivi da 0 a 7 e le righe con le lettere dell'alfabeto da "A" a "H".

I caratteri sono memorizzati nella zona ROM chiamata generatore di caratteri. Ogni carattere occupa 8 byte della ROM.

Per capire come un carattere venga memorizzato, osserviamo la figura 1 e

considieriamo gli 8 dot presenti nella riga indicata con "A". A ogni dot viene associato un numero che è la potenza di 2 con esponente eguale al numero della colonna in cui esso è presente.

Se, ad esempio, vogliamo segnalare l'accensione del dot posto nella riga "A" e in colonna 0, dobbiamo mettere nel byte corrispondente il numero 2 elevato a 0, cioè 1.

Nell'esempio riportato in figura 1, sono accesi i dot corrispondenti alle colonne 3 e 4 della riga "A", per cui il numero da riportare è 2 elevato a 3 sommato a 2 elevato a 4, cioè 24.

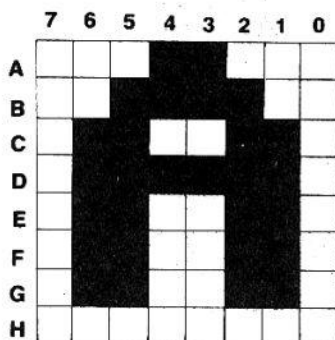



Figura 1

Analogamente, per le altre righe, si avrà:

riga A = 24
riga B = 60
riga C = 102
riga D = 126
riga E = 102
riga F = 102
riga G = 102
riga H = 0

I caratteri che corrispondono ai tasti del calcolatore sono memorizzati nella ROM a partire dalla locazione di memoria 53444.

Da questo, come già detto, si ha che dalla locazione 53444 alla 53451 (8 byte) sono riportati i numeri che corrispondono al carattere chiocciola, da 53452 a 53459 (altri 8 byte) sono riportati i valori corrispondenti alla lettera A e così via per le altre lettere in ordine alfabetico.

Controllando, mediante PEEK, i valori di dette locazioni, si ha la sorpresa di non trovare i valori citati perchè tale parte di ROM è "coperta" da memoria RAM.

Programmazione dei caratteri

Ora che è stato descritto come il C-16 memorizza i caratteri, sarà facile comprendere come si possano definire i caratteri personalizzati.

Occorre innanzitutto dire che nelle locazioni di memoria 65298 e 65299 sono presenti i cosiddetti **puntatori del generatore di caratteri,** cioè due numeri che indicano al microprocessore la locazione di memoria da cui inizia la memorizzazione dei caratteri da sviluppare.

In tali locazioni sono presenti, al momento dell'accensione del calcolatore, due valori che corrispondono appunto alla locazione di memoria 53444. Ciò equivale a dire che:

```
PRINT PEEK(65298)+
PEEK(65299)*256
```

fornisce come risultato 53444.

Se, al posto dei numeri che corrispondono a tale locazione poniamo, mediante istruzioni POKE, altri valori, il microprocessore "punterà" ad altre locazioni di memoria per leggere i caratteri.

In tal modo possiamo definire un'altra area, solitamente scelta alla fine dell'area Basic, come zona per "depositare" i caratteri personalizzati.

Nel C-16 senza espansione di memoria è preferibile utilizzare i valori 0 e 48 per le locazioni 65298 e 65299 rispettivamente, che corrisponde a collocare i nuovi caratteri a partire dalla locazione di memoria che si ricava con la seguente formula:

48x256 + 0 = 12288.

Per esemplificare il metodo utilizzato per programmare i caratteri, si riporta un semplice programma in Basic (figura 2) che farà apparire sullo schermo un piccolo volto sorridente.

Dopo aver digitato il programma e dato il RUN, si noterà che premendo il tasto corrispondente alla lettera A apparirà la faccia sorridente, mentre premendo gli altri tasti appariranno caratteri casuali proprio perchè non sono stati definiti. Per "ritornare" a rendere leggibile la tastiera, dovete nell'ordine:

- tener premuto il tasto Run/stop;
- premere il tasto laterale di Reset (**senza** rilasciare Run/stop);
- rilasciare il tasto di Reset. A questo punto, sul video, compare un messaggio "strano" (è il monitor del C-16);
- rilasciare il tasto Run/stop;
- premere il tasto "X" e, subito dopo, Return. Digitando LIST il programma di prima ritornerà sul video, perfettamente leggibile.

Per facilitare la preparazione dei nuovi caratteri, abbiamo preparato un breve programma Basic che semplificherà notevolmente il lavoro. Dopo averlo digitato e dato il RUN, apparirà sullo schermo una griglia che rappresenta un carattere ingrandito. Con le frecce riportate sulla tastiera si può spostare il cursore, mentre premendo il tasto relativo all'asterisco si annerirà la casella corrispondente a un dot.

Quando avrete costruito i caratteri, premendo il tasto Return saranno disponibili due opzioni: potrete visualizzare sullo schermo i caratteri programmati o ottenere i valori numerici da inserire in memoria per realizzare i caratteri. Per eseguire quest'ultima operazione può essere utilizzato il programma riportato in figura 2.

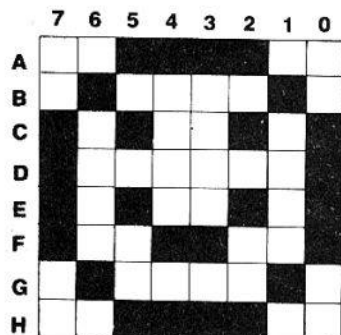



Figura 2

Un'ultima nota: se, durante l'uso di caratteri personalizzati, si commette un errore di sintassi (o di altro tipo), lo schermo va in tilt ed è possibile recuperare il programma solo ricorrendo alla procedura prima descritta (uso dei tasti Run/stop e Reset).

Come utilizzare i caratteri

L'impiego dei caratteri personalizzati può essere utile in numerosi casi, ad esempio:

- per rendere più interessante la presentazione dei programmi, facendo apparire caratteri particolari;
- per costruire alfabeti di lingue straniere (come quella greca);
- per costruire nuovi caratteri semigrafici da utilizzare nei videogiochi.

Per dimostrare quest'ultima opportunità, vi proponiamo un semplice videogioco (Artigliere) il cui listato è riportato in queste pagine.

```
1 REM TRASFORMAZIONE
2 REM DEL CARATTERE "A"
3 REM IN UN VOLTO SORRIDENTE
4 :
5 REM SOLO PER C-16
6 :
10 FOR I=12296 TO 12303:READA:POKEI,A:NEXT:REM LETTURA DEI DATA
20 DATA 60,66,165,129,165,153,66,60
30 PRINT CHR$(147): REM CALCELLAZIONE SCHERMO
40 POKE 65298,0:POKE 65299,48:REM SPOSTA PUNTATORI
```

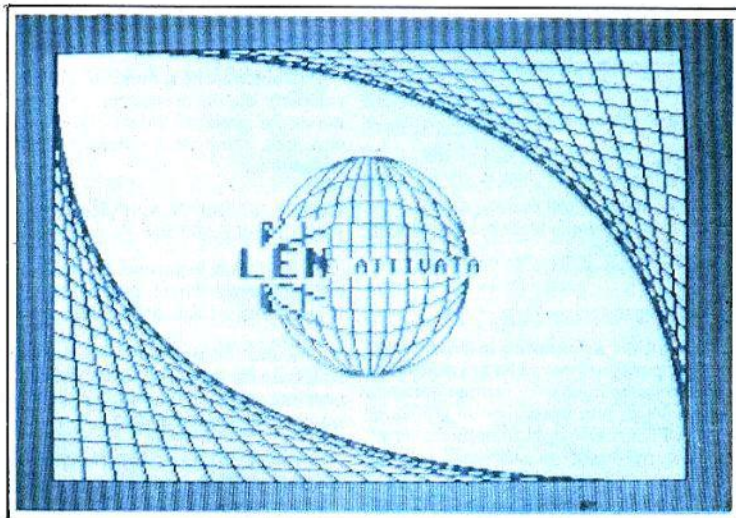
```
3 REM PROGRAMMA PER
4 REM LA DEFINIZIONE
5 REM DEI CARATTERI
6 REM -SOLO PER C-16
7 :
10 GOSUB570:PRINT
20 INPUT"QUANTI CARATT. VUOI PROGRAMMARE (MAX 10)";N:IFN<1ORN>10THEN20
30 DIMA(N*8):REM DIMENSIONAMENTO MATRICE
40 FORZ=0TON-1
50 GOSUB570
60 PRINTCHR$(17)"*=ACCENDE UN PIXEL"
70 PRINT"RETURN=CALCOLA I DATI"
80 PRINT"CARATTERE N°";Z+1;CHR$(145)
90 PRINTSPC(15)CHR$(27)"T"
100 PRINTCHR$(18)"..........":FORT=0TO7:PRINTCHR$(18)"."SPC(8)".":NEXT
110 PRINTCHR$(18)".........."
120 A=3368:B=0
130 REM  CONTROLLO DEL LAMPEGGIAMENTO DEL CURSORE
140 IFB=1THENPOKEA,PEEK(A)OR128:B=0:GOTO170
150 IFB=0THENPOKEA,PEEK(A)AND127:B=1
160 REM CONTROLLO DELLO SPOSTAMENTO DEL CURSORE
170 FORT=0TO30:GETA$:IFA$=""THENNEXT
180 IFA$=CHR$(145)ANDPEEK(A-40)<>160THENGOSUB250:A=A-40:GOTO140
190 IFA$=CHR$(17)ANDPEEK(A+40)<>160THENGOSUB250:A=A+40:GOTO140
200 IFA$=CHR$(157)ANDPEEK(A-1)<>160THENGOSUB250:A=A-1:GOTO140
210 IFA$=CHR$(29)ANDPEEK(A+1)<>160THENGOSUB250:A=A+1:GOTO140
220 IFA$="*"THENPOKEA,42:REM STAMPA DELL'ASTERISCO
230 IFA$=CHR$(13)THEN260:REM EVENTUALE PASSAGGIO ALLA SUBROUTINE 260
240 GOTO140
250 POKEA,PEEK(A)AND127:RETURN
260 GOSUB250:C=0:FORA=0TO7:FORB=0TO7:REM CALCOLO DEI DATA
270 IFPEEK(3368+A*40+7-B)=32THEN280:ELSEC=C+2↑B
280 NEXT:D(A)=C:C=0:NEXT
290 FORT=0TO7:A(W)=D(T):W=W+1:NEXT:PRINTCHR$(27)"N":NEXT
300 GOSUB570
310 PRINT:PRINT"1) PER VEDERE I DATI DA INSERIRE NEL    PROGRAMMA"
320 PRINT:PRINT"2) PER VEDERE I CARATTERI PROGRAMMATI"
```

```
330 REM ROUTINE DI INPUT CONTROLLATO
340 GETA$:IFA$="1"THEN380
350 IFA$="2"THEN420
360 GOTO340
370 GOSUB570
380 PRINT:PRINT"DATA DA INSERIRE NEL PROGRAMMA:"
390 W=0:FORT=1TON:FORH=0TO7:PRINTA(W);:W=W+1:NEXT:PRINT:NEXT
400 PRINT:PRINT"PREMI UN TASTO":WAIT198,32:GOTO310
410 GOSUB570
420 PRINT:PRINT"PER VEDERE I CARATTERI"
430 PRINT"PREMI I SEGUENTI TASTI: ";
440 FORT=1TON:PRINTCHR$(63+T);CHR$(32);:NEXT:PRINT
450 PRINT:PRINT"QUANDO HAI FINITO DI VEDERE I CARATTERI"
460 PRINT"PREMI IL TASTO RETURN"
470 PRINT
480 PRINTSPC(5)"PREMI UN TASTO"
490 REM SCRITTURA DEI DATA IN MEMORIA
500 WAIT198,32:FORT=0TO7:POKE12544+T,0:NEXT:PRINTCHR$(147)
510 FORT=0TON*8:POKE12288+T,A(T):NEXT
520 POKE65298,0:POKE65299,48
530 GETA$
540 IFA$=CHR$(13)THEN560
550 PRINTA$;:GOTO530
560 POKE65298,196:POKE65299,208:GOTO300:REM SPOSTAMENTO DEI PUNTATORI
570 PRINTCHR$(147)SPC(4)"PROGRAMMA PER LA COSTRUZIONE"
580 PRINTSPC(10)"DEI CARATTERI":RETURN
590 END
```

```
3 REM L'ARTIGLIERE
4 REM -GIOCO PER C-16-
5 REM SOFTWARE DI
7 REM FEDERICO CANETTA
9 :
10 PRINTCHR$(147)CHR$(31)SPC(13)"L'ARTIGLIERE"
20 PRINT
30 PRINTCHR$(144);"CERCATE DI COLPIRE IL BERSAGLIO CON IL"
40 PRINT"CANNONE. USATE LE FRECCETTE PER MUOVERE"
50 PRINT"IL CANNONE E LO SPAZIO PER SPARARE.":PRINT
60 PRINTSPC(5)"PREMI UN TASTO PER INIZIARE"
70 WAIT198,32
80 PRINTCHR$(147)
90 FORT=0TO31:READA:POKE12288+T,A:NEXT
100 POKE65298,0:POKE65299,48:VOL8
110 A=3072:B=4052:POKEB,1
120 FORT=0TO1000:POKE3072+T,0:POKE3071+T,3:SOUND3,T,2
130 GETA$:IFA$=CHR$(157)ANDB>4032THENPOKEB,3:B=B-1:POKEB,1
140 IFA$=CHR$(29)ANDB<4071THENPOKEB,3:B=B+1:POKEB,1
150 IFA$=" "ANDD=0THEND=1:C=B-40
160 IFD=1THENPOKEC,3:C=C-40:POKEC,2:SOUND1,C-3072,2:IFC<3152THENPOKEC,3:D=0
170 IFPEEK(3072+40+T)=2THENP=P+10:POKEC,3:POKE3072+T,3:T=0:D=0
180 NEXT
190 POKE65298,196:POKE65299,208:PRINTCHR$(147)
200 PRINT"HAI FATTO";PU;"PUNTI"
210 PRINT:PRINT"VUOI GIOCARE ANCORA (S/N) ?"
220 GETA$:IFA$="S"THENRUN
230 IFA$="N"THENEND
240 GOTO220
250 DATA 60,66,165,129,165,153,66,60
260 DATA 24,24,24,24,24,60,255,255
270 DATA 24,24,24,24,24,24,24,24
280 DATA 0,0,0,0,0,0,0,0
290 END
```

Lente di ingrandimento per Commodore 64

di Fabio Sorgato

Aggiungiamo un nuovo comando alle famose routine grafiche tridimensionali: simulando la presenza di una lente sullo schermo Hi-Res del C-64 si ottengono effetti stupefacenti.

La lente è un dispositivo ottico sul quale sono ricavate superfici curve, di solito sferiche, in modo che la radiazione luminosa subisca una certa deviazione attravesando la lente stessa.

Una caratteristica delle lenti è quella di avere un asse ottico, cioè un punto che, attraversato dai raggi luminosi, non conferisce loro nessuna deviazione.

E' possibile realizzare lenti convergenti, tali, cioè, che attraversate da un fascio di raggi paralleli, li concentrano in un dato punto detto fuoco; oppure lenti divergenti che formano un fuoco virtuale, che in realtà non esiste, creato dai prolungamenti dei raggi all'indietro.

Le lenti convergenti sono quelle usate per la costruzione di strumenti, quali microscopi e canocchiali, mentre le lenti divergenti sono in genere usate in unione alle precedenti per svolgere altre "funzioni" ottiche.

La lente qui simulata è proprio una lente convergente già messa "a fuoco" in modo da ottenere l'immagine nitida. Questa, come già detto, è rappresentata dai disegni, grafici e, insomma, tutto ciò che è presente sulla pagina in alta risoluzione utilizzata normalmente nelle routine grafiche di D.Toma.

Come si usa

Dopo avere digitato e registrato a parte il programma qui pubblicato, si devono caricare le routines grafiche (apparse sul N.14 di C.C.C.), attivarle, caricare il programma digitato e dare il RUN. Se non avrete commesso errori di trascrizione vedrete apparire una dimostrazione delle possibilità offerte dalla lente.

Per manovrarla è stato aggiunto un nuovo comando che, come tutti gli altri, deve essere preceduto dal carattere di freccia a sinistra (primo tasto in alto a sinistra):

LENS X,Y,C

Il valore X indica la coordinata orizzontale del centro della lente (asse ottico). I valori assegnabili sono compresi tra 0 e 511. Il valore per porre l'asse ottico al centro dello schermo è 148.

Il valore Y indica le coordinate verticali dell'asse ottico della lente. I valori possono variare tra 0 e 255 e quello di centro schermo vale 89.

Il parametro C indica il colore dei punti ingranditi e può avere un valore compreso tra 0 e 255: si consiglia di usare lo stesso colore scelto per il disegno. La tabella dei colori è quella "solita" degli sprites.

Inoltre valgono sempre le regole del comando "$", vale a dire la riassegnazio-

ne del valore precedente in posizione analoga.

Se per esempio eseguiamo:
LENS 100,100,0
e poi:
LENS $,50,$
il comando eseguito sarà:
LENS 100,50,0

Chiaramente anche se il comando precedente sarà diverso, per esempio:
DRAW 10,20,0
il risultato non varierà.

Ricordiamo però che per mettere al centro della lente un punto con la coordinata Z uguale a zero (la lente non lavora in 3-D), non possiamo porre gli stessi valori delle coordinate del punto ma dobbiamo usare le seguenti formule:
per il valore X:

148+coordinate X del punto

per il valore Y:

89-coordinate Y del punto

Per disattivare la lente basterà semplicente eseguire:
POKE 53269,0
oppure:
LENS 400,0,0

Un'ultima avvertenza: la lente è stata creata per funzionare con la grafica monocromatica in alta risoluzione quindi si consiglia di non usarla con la grafica in MULTICOLOR oppure in modo testo: si otterrebbero, in tal caso, effetti imprevedibili.

Come funziona

La lente è uno sprite, in cui vengono trasferiti i dati, con le coordinate "X" e "Y" raddoppiate, posto nella zona di schermo da cui vengono prelevati i dati stessi.

Le difficoltà per creare una routine che faccia questo tipo di lavoro non sono notevoli, ma non vanno tuttavia sottovalutate.

Ora l'analizzeremo e cercheremo di capire il criterio con cui è stata creata e, se siete un po' esperti, potrete riprodurre la lente in BASIC.

Il problema che si presenta è quello di calcolare la zona di memoria da cui devono essere prelevati i dati, che si risolve con una semplice (o quasi!) formula matematica:

L=START+40*(Y AND 504)+Y AND 7 + X AND 504

Dove L è la locazione di memoria da cui è prelevato il byte, mentre START è la locazione di inizio della pagina grafica.

Ora che abbiamo il dato, lo dobbiamo trasferire nello sprite badando che la configurazione di uno sprite è diversa da quella della pagina grafica.

I problemi maggiori sorgono quando si deve creare un movimento punto-punto all'interno della lente e ciò si deve risolvere con delle "rotazioni" binarie, in cui sconsiglio di cimentarvi se non siete veramente esperti.

Solo se, infatti, riuscirete a riprodurla con un movimento di 8 bit (un byte) come passo, potrete considerarvi esperti conoscitori della programmazione BASIC.

Per muovere lo sprite nella posizione esatta, dobbiamo ricordare che il punto sotto l'asse focale (il centro dello sprite) non deve apparire spostato dalla posizione reale.

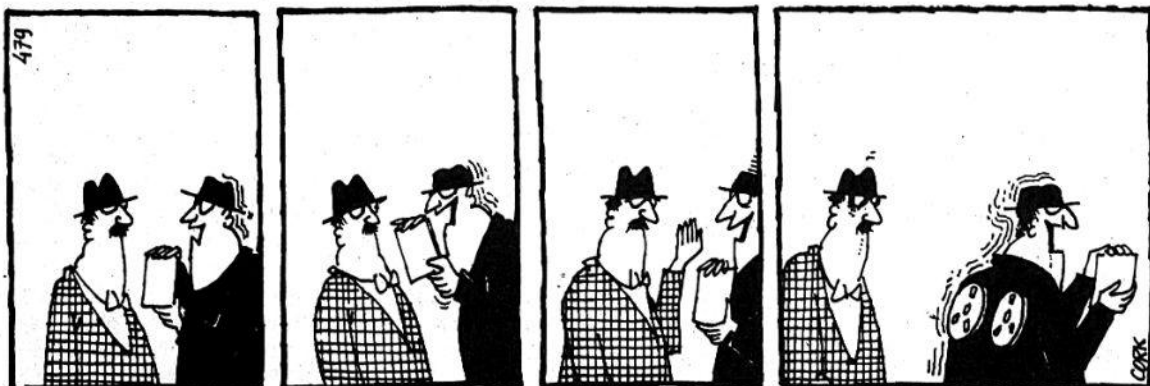
Se, quindi, poniamo lo sprite (raddoppiato in dimensioni e con precedenza nella sovrapposizione sprite-fondo) nella parte desiderata, osserveremo una sovrapposizione di punti con il disegno sottostante perchè, come ben sappiamo, uno sprite è trasparente.

La soluzione ideale è porre un secondo sprite "pieno", cioè completamente colorato con il colore del fondo dello schermo, e con precedenza di visualizzazione inferiore rispetto a quello contenente i dati della lente. Il secondo sprite possiede, ovviamente, le stesse coordinate del primo.

Ricordiamo che la precedenza di visualizzazione dello sprite è decisa in base al numero dello sprite stesso: minore è il numero, maggiore è la precedenza.

Questi sono i criteri usati per la creazione della lente. Tocca a voi, ora, cimentarvi nelle infinite applicazioni possibili.

```
 10 REM  SIMULAZIONE DI UNA
 20 REM  LENTE D'INGRANDIMENTO
 30 REM  CON LE  ROUTINES
 40 REM  GRAFICHE DI D.TOMA
 50 REM  PER COMMODORE 64
 60 REM  PUBBLICATE SUL NUMERO 14
 70 :
 80 REM  SOFTWARE BY FABIO SORGATO
 95 :
100 PRINTCHR$(147):PRINT:PRINT"ATT
    ENDERE..."
105 FOR K=0 TO 307:READ A:PRINTCHR
    $(19)307-K;CHR$(157)CHR$(32)
110 POKE 49152+8*256+K,A
120 CK=CK+A:NEXT
130 IF CK=38913 THEN 150
140 PRINT"ERRORE NEI DATI":END
141 :
150 PRINTCHR$(147):POKE  51164,002
160 POKE  51165,195
170 POKE  51166,083
180 POKE  50190,000
190 POKE  50191,200
200 POKE  51051,010
201 :
210 ←CLEAR:←GRAF1,0:←COL OR 1
220 FOR K=0 TO 200 STEP 10
230 ←DRAW-160,100-K,0,K*1.6-160,-1
    00,0
240 ←DRAWK*1.6-161,99,0,159,99-K,0
250 NEXT
251 FOR K=0 TO 50 STEP 10:←CIRCLE0
    ,0,0,K,50:NEXT
252 FOR K=0 TO 40 STEP 10:←CIRCLE0
    ,0,0,50,K:NEXT
260 POKE 56334,0:POKE 1,51
```

```
270 A$="LENTE ATTIVATA"
280 FOR K=1 TO LEN(A$):V=ASC(MID$(
    A$,K,1))-64:IF V<0 THEN V=V+64
285 P=V*8+53248
290 L=(K-1)*8+320*12+8*13+57344
300 FOR J=0 TO 7:POKE L+J,PEEK(P+J
    ):NEXT:NEXT
310 POKE 1,55:POKE 56334,1
320 FOR J=1 TO 5
330 FOR X=0 TO 320:←LENS X,90,0:NE
    XT:POKE 53269,0
335 FOR X=0 TO 200:←LENS 148,X,0:N
    EXT:POKE 53269,0:NEXTJ
340 FOR K=1 TO 600:NEXT:←TEXT6,14:
    END
350 :
1000 DATA  032,143,197,173,208,002
1010 DATA  133,087,173,209,002,041
1020 DATA  254,240,005,162,014,108
1030 DATA  000,003,173,209,002,133
1040 DATA  088,173,210,002,133,089
1050 DATA  173,211,002,208,239,173
1060 DATA  212,002,133,091,173,213
1070 DATA  002,208,229,173,000,207
1080 DATA  041,015,133,090,169,040
1090 DATA  141,248,207,169,041,141
1100 DATA  249,207,169,003,141,021
1110 DATA  208,141,023,208,141,029
1120 DATA  208,165,090,141,040,208
1130 DATA  165,091,141,039,208,120
1140 DATA  169,053,133,001,169,000
1150 DATA  133,094,133,002,165,089
1160 DATA  074,074,074,010,133,251
1170 DATA  169,160,133,252,032,026
1180 DATA  201,165,087,041,007,170
1190 DATA  165,087,041,248,133,090
1200 DATA  165,089,041,007,005,090
1210 DATA  024,101,254,144,002,230
1220 DATA  255,133,254,165,255,024
1230 DATA  101,088,105,224,133,255
1240 DATA  160,000,177,254,153,090
1250 DATA  000,165,254,024,105,007
1260 DATA  144,002,230,255,133,254
1270 DATA  200,192,004,208,235,138
1280 DATA  240,011,006,093,038,092
1290 DATA  038,091,038,090,202,208
1300 DATA  245,164,094,162,000,181
1310 DATA  090,153,192,003,230,094
1320 DATA  200,232,224,003,208,243
1330 DATA  230,089,230,002,165,002
1340 DATA  201,021,208,142,160,000
1350 DATA  165,087,024,105,012,144
1360 DATA  002,160,003,141,000,208
1370 DATA  141,002,208,165,089,024
1380 DATA  105,018,141,001,208,141
1390 DATA  003,208,165,088,240,002
1400 DATA  160,003,132,251,173,016
1410 DATA  208,041,252,005,251,141
1420 DATA  016,208,162,064,160,255
1430 DATA  189,191,003,157,255,201
1440 DATA  152,157,063,202,202,208
1450 DATA  243,169,055,133,001,169
1460 DATA  096,141,065,003,088,096
1470 DATA  169,000,162,008,010,038
1480 DATA  251,144,007,024,101,252
1490 DATA  144,002,230,251,202,208
1500 DATA  241,133,254,165,251,133
1510 DATA  255,096
```

C 16, ed è subito grafica

di **Michele Maggi**

Una classica applicazione (disegno di istogrammi) per i computer dotati di specifiche istruzioni grafiche. E, dato che ci siamo, qualche spiegazione sull'uso corretto di alcune di queste.

Il C16, il più piccolo computer della famiglia Commodore, a dispetto del suo basso costo si rivela estremamente potente e versatile soprattutto dal punto di vista della grafica.

Il Basic 3.5 infatti contiene vari comandi che ne rendono molto semplice l'utilizzo. Le istruzioni per disegnare sono:

BOX, CHAR, CIRCLE, DRAW, GRAPHIC, LOCATE, PAINT.

E' interessante notare che vi sono ben quattro tipi di alta risoluzione in modo da venire incontro alle più disparate esigenze. E' possibile, tra l'altro, dividere lo schermo in due parti (parte grafica e parte testo) e ancora, la parte grafica può essere utilizzata in modo multicolor o normale.

Cosa significa tutto ciò?

In primo luogo vediamo che, essendo possibile dividere (in inglese, split) lo schermo in due parti, si potrà, ad esempio, scrivere una routine di input nella parte testo dello schermo senza compromettere la visualizzazione della pagina grafica. Nel programma di queste pagine viene appunto sfruttata questa tecnica.

Per ciò che riguarda la pagina grafica, a parte la possibilità di visualizzarla assieme alla pagina testo, esiste la possibilità di utilizzarla in modo normale o multicolor.

Alta risoluzione normale

Prima di tutto, che significa alta risoluzione (detta anche Hi-Res)?.

Sappiamo che lo schermo normale del C16 contiene 25 righe di 40 caratteri ciascuna per un totale, quindi, di 1000 locazioni (o celle) di schermo. Però sappiamo anche che ogni carattere è formato da una matrice di 8X8 punti. Avremo dunque 200 file composte da 320 punti ciascuna per un totale di 64000 punti (25 x 40 x 8 x 8).

Il nostro computer è appunto in grado di controllare singolarmente ognuno di questi punti, detti in inglese DOT.

Questo fatto permette sicuramente di creare disegni o grafici enormemente più precisi di quelli in bassa risoluzione. Potremo quindi utilizzare le istruzioni grafiche per gestire un numero notevole di figure: disegnare cerchi, rettangoli, poligoni e numerosissime altre forme.

Vediamo come:

Il primo passo verso un grafico Hi-Res si compie mediante l'istruzione GRAPHIC. La sua sintassi è la seguente:
GRAPHIC modo , cancellazione

Esistono 5 tipi di modalità che includono, o meno, la possibilità di cancellare lo schermo.

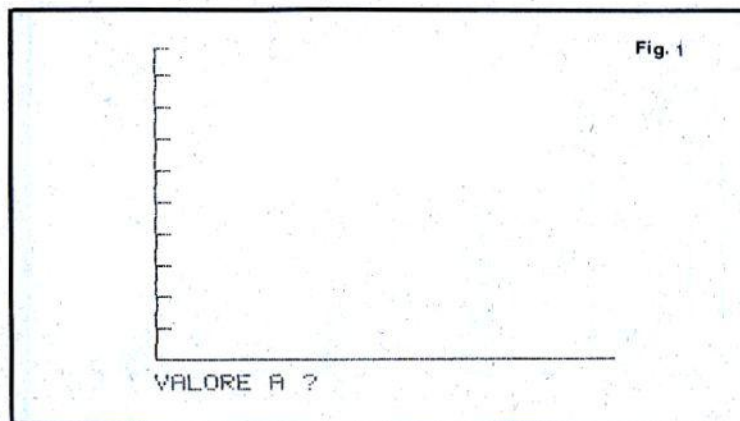
Istruzione	*Modo*
GRAPHIC 0	Testo
GRAPHIC 1	Alta risoluzione
GRAPHIC 2	Alta risoluz. +testo
GRAPHIC 3	Multicolor
GRAPHIC 4	Multicolor + testo

Cancellazione dello schermo grafico

Per cancellare lo schermo grafico sarà sufficiente aggiungere "virgola uno" (,1) ad una delle istuzioni GRAPHIC. Per esempio, GRAPHIC 1,1 porta lo schermo in Alta Risoluzione cancellando ogni figura eventualmente presente.

GRAPHIC 2,1 porterà lo schermo in alta risoluzione + testo e così via.

Va da sè che l'istruzione GRAPHIC da sola serva a ben poco in quanto, è vero che "ci porta" in alta risoluzione, ma è anche vero che, se non usiamo almeno una delle altre istruzioni, non vedremo nulla di interessante.

Un'ultima considerazione prima di passare a studiare le istruzioni in dettaglio: quando si utilizza la pagina grafica, il sistema riserva per sè un'area di ben 10Kbyte (diecimila locazioni circa), per cui se sottraiamo questi 10K dai 12K del nostro C16, notiamo che ci restano a disposizione poco più di 2K. Ciò significa che dovremo essere piuttosto stringati nella stesura dei programmi che utilizzano la grafica anche pperchè, oltre allo

spazio occupato dal programma in sè, dobbiamo ricordarci che quarto, una volta lanciato, potrà via altra memoria soparttutto se contiene vettori o matrici.

Esiste comunque l'istruzione GRAPHIC CLR che permette di riutilizzare i 10K qualora non serva più lo schermo grafico.

Uso dell'istruzione "LOCATE"

Quando si lavora in modo testo, siamo abituati a vedere lampeggiare il cursore che segnala esattamente in quale punto dello schermo ci troviamo.

Per il nostro cursore esisteranno, dunque, esattamente 1000 posizioni possibili. In alta risoluzione, ovviamente, cambierà qualche cosa: il normale cursore non esiste più (se non nella parte testo, ma questo è un altro discorso) e viene sostituito, nella pagina grafica, dal Cursore Pixel (CP) che potrà assumere non 1000 ma 64000 posizioni.

L'istruzione LOCATE permette di individuare, mediante due coordinate, uno qualsiasi di questi punti. Ad esempio LOCATE 160,100 posizionerà il CP al centro dello schermo grafico.

A differenza del cursore normale, il CP è invisibile e, per visualizzarlo, dovremo ricorrere ad una delle istruzioni che permettono effettivamente di disegnare.

Vediamone una.

L'istruzione BOX

Questa istruzione, senz'altro una delle più utili fra quelle che abbiamo a disposizione, consente di disegnare forme rettangolari, vuote o piene, con la possibilità di ruotarle di un certo numero di gradi.

Il suo tipico formato è il seguente:

BOX 1,X,Y,X1,Y1

in cui: X e Y rappresentano le coordinate di partenza del riquadro, mentre X1 e Y1 sono le coordinate di arrivo.

Il numero 1 che appare subito dopo BOX significa che vogliamo disegnare e non cancellare: se lo sostituiamo con uno 0 potremo cancellare il riquadro precedentemente disegnato.

Se invece di un riquadro vuoto, vogliamo disegnare un riquadro pieno, basterà aggiungere "virgola virgola uno" (,,1) in coda all'istruzione stessa:

BOX 1,X,Y,X1,Y1,,1

Se vogliamo ruotare il riquadro di un certo numero di gradi, basterà aggiungere in coda all'istruzione ANG; dove ANG è il numero di gradi (es. 45).

Provate ora a digitare il seguente programmino:

```
10 GRAPHIC 1,1
20 BOX 1,100,100,150,50
30 GETKEYA$:GRAPHIC 0
```

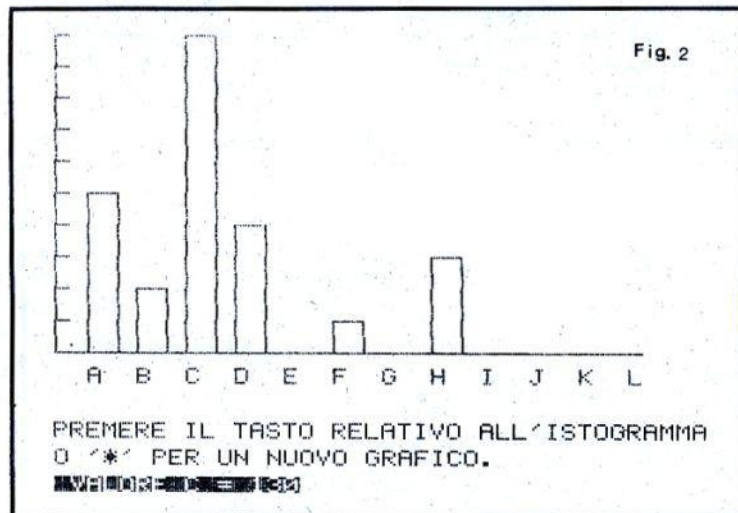
La riga 10 ci porta in alta risoluzione a schermo intero; 20 traccia il riquadro e la 30 attende la pressione di un tasto per riportare tutto alla normalità.

Provate ora a modificare la riga 20 aggiungendo ,,1 alla fine dell'istruzione 'BOX'

```
20 BOX 1,100,100,150,50,,1
```

Allo stesso punto di prima viene ora tracciato un riquadro pieno.

Se proviamo ora a modificare la riga 20 in questo modo:

```
20 BOX 1,100,100,150,50,45
```

otterremo un riquadro ruotato di 45 gradi.

Non abbiate paura di cambiare a vostro piacimento i vari parametri dell'istruzione BOX in modo da capire esattamente come funziona.

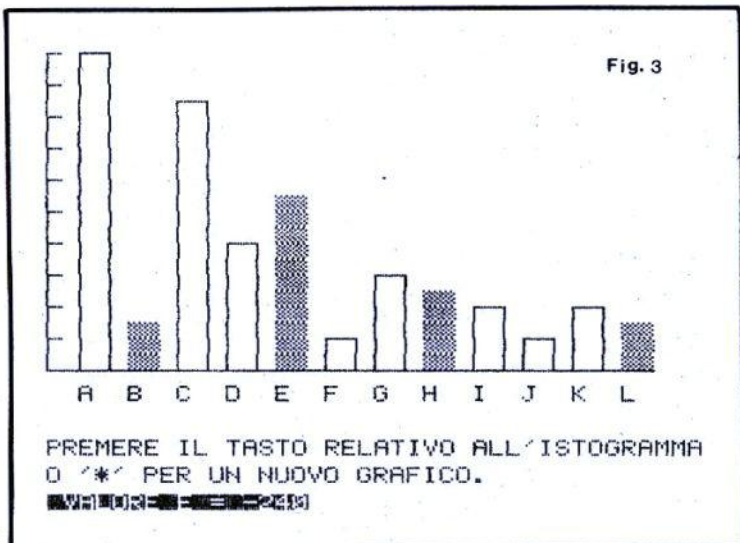
Un consiglio: non esagerate con le modifiche ai parametri per non incorrere nel fatidico ILLEGAL QUANTITY ERROR.

Come applicazione del comando BOX analizziamo insieme il programma ISTOGRAMMI.

Istogrammi per C16

nnanzittutto, che cos'è e a cosa serve un istogramma?

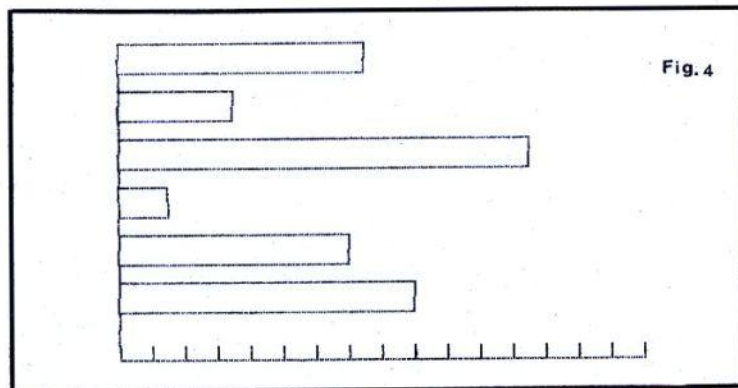
Un istogramma è un pratico e veloce modo per visualizzare una tabella di valori, come ad esempio le vendite realizzate in un periodo di tempo definito (es. anno). Spesso, infatti, invece di confrontare dei numeri fra di loro, è meglio visualizzare la loro "dimensione" grafica, così da poter capire, con una semplice occhiata, quale valore sia più piccolo o più grande.

Un programma che genera istogrammi si rivela utile in numerose applicazioni: può essere adoperato per analizzare l'andamento dei costi o delle vendite di un'azienda oppure per esaminare l'incidenza delle entrate e delle uscite, o ancora per avere una visualizzazione grafica dell'andamento scolastico.

Nella fase di creazione di un programma che genera istogrammi, esistono essenzialmente due problemi e, conseguentemente, due scelte da effettuare.

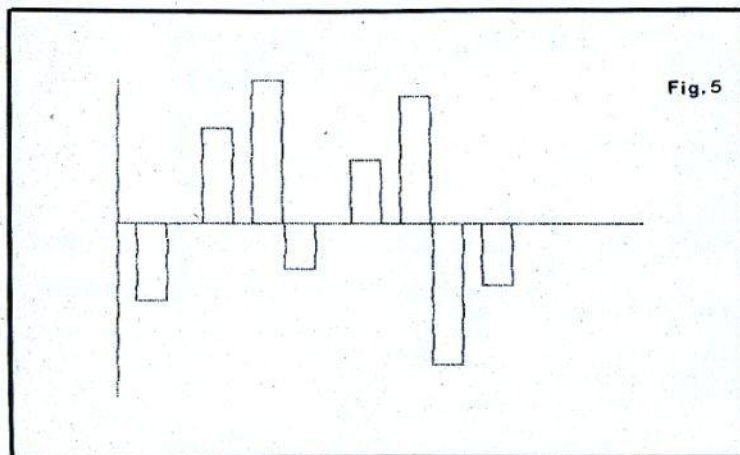
Per prima cosa bisogna stabilire se i nostri istogrammi dovranno svilupparsi in senso verticale oppure orizzontale; il secondo problema è invece attinente alla rappresentazione dei dati negativi.

Dato che lo schermo utilizzabile è più largo (40 colonne, 320 dots), che lungo (25 righe, 200 dots), se avessimo deciso di adottare la forma orizzontale avremmo avuto forse una maggior precisione nel disegno dell'istogramma (sarebbe stato più lungo) ma avremmo potuto visualizzare un minor numero di dati (vedi fig. 3). Si è quindi deciso di adottare la forma verticale che consente di visualiz zare un numero praticamente doppio di istogrammi (fig.2).

A tutto ciò va aggiunto che, con l'impostazione di alta risoluzione + testo (fig.1), si sarebbe persa per il disegno anche quella parte riservata al testo.

Una volta operata la scelta relativa al senso in cui deve svilupparsi l'istogramma, possiamo occuparci del secondo problema.

La rappresentazione dei dati negativi

Per risolvere questo problema si adottano, di norma, tre soluzioni:

- creare un programma che non accetta dati negativi;
- visualizzare i dati negativi al di sotto della linea di mezzo (fig.5);
- visualizzare i dati negativi con riquadri pieni al posto di quelli vuoti in modo da differenziarli tra loro (fig.4).

Considerato che spesso, in una tabella di valori, sono presenti valori negativi, è stata scartata la prima soluzione, perchè comporterebbe una minor versatilità del programma stesso.

La seconda soluzione, d'altronde, avrebbe comportato l'utilizzo di istogrammi più ridotti e quindi meno precisi. La forma di istogramma verticale con visualizzazione inversa dei dati negativi va considerata, tutto sommato, un ottimo compromesso.

Come gira il programma

Vediamo ora cosa fa effettivamente il nostro programma.

Una volta dato il Run, ci troviamo subito in modo testo + grafica con il disegno degli assi cartesiani e con la richiesta in INPUT del primo dato (fig.1). Una volta inseriti tutti i dati, il programma visualizzerà gli istogrammi al di sotto dei quali si troverà una lettera dell'alfabeto (A-L). Premendo il tasto corrispondente otterremo il valore dell'istogramma relativo (fig.2 e 4).

Occorre notare che non è necessario inserire 12 valori. Se, infatti, alla richiesta di inserimento rispondiamo semplicemente col tasto Return, l'istogramma passa automaticamente a zero e, di conseguenza, non verrà visualizzato.

Se invece di uno dei tasti corrispondenti a un istogramma, viene premuto l'asterisco (*), il programma riparte nuovamente mantenendo però i primi valori dati in INPUT. Questo per facilitare un tipo di analisi un po' più complessa. Per cambiare i dati sarà sufficiente scrivere "sopra" quelli vecchi e per azzerarli bisognerà coprire di zeri (oppure cancellare con la barra spaziatrice) il dato precedente.

Come funziona il programma

A parte le linee 10-99 che possiamo ovviamente trascurare, il programma si struttura in 17 righe numerate da 100 a 260.

La riga 100 porta il C16 in modo alta risoluzione t testo, dimensiona i tre vettori (F,H,K,) e assegna a V$ la parola 'valore'.

La riga 110 disegna gli assi cartesiani.

La 120 setta la variabile A=19 (punto di partenza del CP per il primo istogramma cfr. istruzione LOCATE); stabilisce il valore del parametro X che sarà usato come Tab nell'istruzione successiva.

La riga 130 provvede all'INPUT dei vari valori e manda alla subroutine 220.

La riga 150 stampa i caratteri da "A" a "L" e stabilisce l'altezza di ogni istogramma.

La riga 160 verifica se il valore è negativo e, in tal caso, setta ad 1 la variabile P che serve per la visualizzazione inversa e prende il dato come valore assoluto.

La riga 170 è il cuore del programma: contiene infatti il comando BOX che disegna effettivamente l'istogramma.

Le righe 180-200 stabiliscono se è stato premuto un tasto relativo al valore di un istogramma oppure se è stato premuto il tasto asterisco (*) che permette di creare un nuovo grafico.

La subroutine 220-260 si occupa del rapporto fra il valore reale dei dati e il valore relativo che essi dovranno assumere per essere visualizzati sullo schermo.

Lista delle variabili

K= Valore che viene dato in INPUT dall'utente per ogni dato.

F(I)= Assume il valore di ogni dato K che viene successivamente utilizzato nel disegno.

H= E' il valore incognito nella seguente proporzione:

H:150=K:MAX

in cui

150 è il numero dei dots in altezza.
K è il dato in INPUT,
MAX è il dato con il più alto valore assoluto.

A= punto di partenza di ogni istogramma.
P= viene settato a 1 se il dato è negativo, questo permette di disegnare ugualmente l'istogramma, al di sopra dell'asse X, ma colorato.
V$= valore
A$= Tasto premuto

```
10 REM COMMODORE C-16
20 :
30 REM ISTOGRAMMI
40 :
50 REM SOFTWARE DI
60 REM MICHELE MAGGI
70 :
99 :
100 GRAPHIC2,1:DIMF(12):DIMH(12):DIMK(12):V$="VALORE "
110 DRAW1,0,150TO320,150:DRAW1,0,0TO0,150:FORT=0TO150STEP15:DRAW1,0,TTO3,T:NEXT
120 A=19:FORI=1TO12:H(I)=0:IFF(I)>=0THENX=9:ELSEX=10
130 PRINTV$;CHR$(64+I)TAB(X)F(I):INPUT"]VALORE■■";K(I):F(I)=K(I):NEXT:GOSUB220
140 PRINTCHR$(147):FORI=1TO20:PRINTCHR$(17);:NEXT
150 FORI=1TO12:PRINT"--"+CHR$(64+I);:NEXT:PRINT:FORI=1TO12:K(I)=150-(H(I)):NEXT
160 LOCATEA,155:FORI=1TO12:IFF(I)<0THENP=1:K(I)=ABS(K(I))
170 BOX1,A,150,A+16,K(I),,P:A=A+24:P=0:NEXT
180 PRINTCHR$(17)"PREMERE IL TASTO RELATIVO ALL'ISTOGRAMMA";
185 PRINT"* PER UN NUOVO GRAFICO."
190 GETKEYA$:IFA$="*"THENGRAPHIC2,1:GOTO110
200 X=ASC(A$):IFX<65ORX>76THEN190
210 PRINTCHR$(18)"VALORE ";A$;" =";F(X-64);
215 PRINTCHR$(146)CHR$(157)"..........."CHR$(145):GOTO190
220 MAX=ABS(K(1))
230 FORI=2TO12
240 IFMAX>=ABS(K(I))THENNEXT:ELSE MAX=ABS(K(I)):GOTO230:NEXT
250 FORI=1TO12:IFK(I)=0THENNEXT:RETURN
260 H(I)=(150*ABS(K(I)))/MAX:NEXT:RETURN
270 END
```

MEZZOGIORNO DI FUOCO

Big Mac (C-16 & PLUS-4)

A dispetto del brevissimo tempo di caricamento (solo 32 giri), il gioco è piuttosto vario ed animato facendo ricorso a più di una tecnica: Alta risoluzione, caratteri ridefiniti, movimento graduale (per singolo dot di schermo) di grosse porzioni del video, creazione del personaggio e del punteggio utilizzando due caratteri sovrapposti.

Una gradevole musichetta fa da sottofondo alla presentazione, mentre effetti sonori di vario tipo vi guidano attraverso le quattro schermate che dovete percorrere avendo a disposizione altrettante "vite". Ogni labirinto deve esser percorso in un tempo stabilito ed è necessario evitare di esser schiacciati da enormi pistoni, colpiti da proiettili o precipitare in pozzi profondi. Ogni schermata rappresenta una delle "stanze" da percorrere per raggiungere la fine del gioco.

E' possibile sospendere temporaneamente il gioco, passare subito alle schermate successive, giocare da soli oppure in due giocatori, usare il joystick oppure la tastiera.

Kane (C-64 C-128)

Ambientato nel Far West, utilizzabile col joystick e caricabile in 112 giri, Kane rappresenta uno di quei giochi in cui l'animazione degli sprite è portata al livello di vero e proprio cartone animato. Il gioco si svolge in quattro schermate, rappresentanti ciascuna una scenografia di grande effetto:

- Nella prima un uomo corre alla postazione designata per sparare a oche selvatiche in volo.

- Nella seconda è necessario, cavalcando con notevole realismo, superare, saltando, i numerosi ostacoli che si incontrano lungo il cammino.

- Nella terza dovete esser lesti ad estrarre la pistola e a sparare prima che pericolosi pistoleri, uscendo allo scoperto in uno scenario da "Mezzogiorno di fuoco", vi colpiscano.

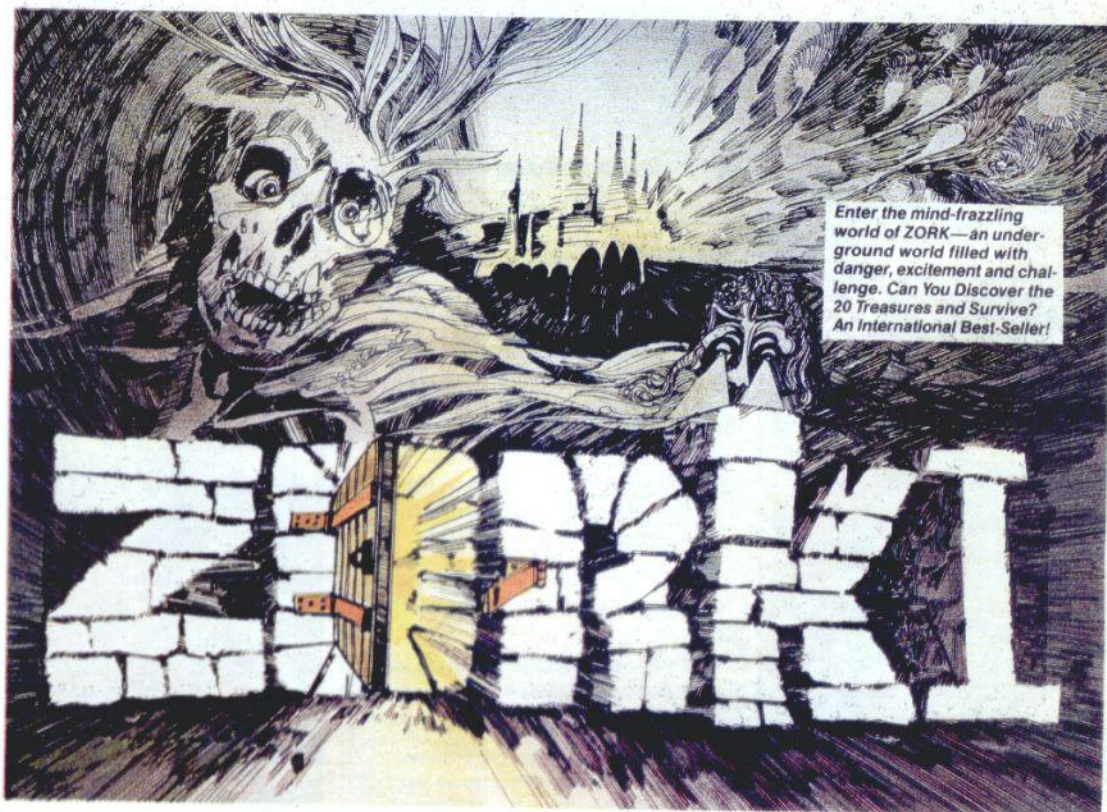
- Nell'ultima schermata dovete raggiungere un treno in movimento cavalcando e superando (era il caso di dirlo?) ostacoli più numerosi e pericolosi di prima.

E' possibile esercitarsi in una delle quattro schermate a scelta, selezionare il livello di difficoltà (1, 2, 3) oppure, ovviamente, richiedere lo svolgimento completo del gioco, fino alla fine.

L'animazione è davvero realistica e paragonabile, per ciò che riguarda l'uomo che corre, al personaggio di "Mission Impossible". La struttura del gioco e la possibilità di selezionare la videata, invece, ricorda "Winter Games".

Zork I, l'impero sotterraneo

di Michele Maggi

Chi è appassionato di Adventures conosce di certo la serie di Zork, gioco ai vertici della sua categoria. Notevole è infatti il coinvolgimento creato nel giocatore che si avventura nel mondo fantastico e sotterraneo.

Prima di proseguire è opportuno spiegare che cosa sia un Adventure a coloro che non ne hanno mai sentito parlare (posto che ne esistano ancora...).

Gli Adventures possono venire definiti in svariati modi. Molti ne parlano come di avventure immaginarie guidate da un computer. Altri li descrivono come libri con il finale "aperto" a significare che ogni giocatore è artefice del proprio destino nella storia raccontata. Altri ancora li giudicano come veri e propri sogni che spesso raggiungono livelli da incubo, e così via...

A parte le varie definizioni che non forniscono una esauriente spiegazione della materia, esistono molti libri che trattano ampiamente l'argomento e insegnano addirittura a realizzare veri e propri Adventures.

Esistono principalmente due categorie di Adventures: quelli puramente descrittivi e quelli cosiddetti semigrafici che, oltre alla descrizione verbale, offrono anche un disegno raffigurante l'ambiente in cui si svolge l'azione.

Non ci dilungheremo nella descrizione delle differenze fra i due tipi perchè risultano ovvie.

In sostanza, un Adventure altro non è che un gioco in cui, invece di colorati e vivaci schermi, vengono visualizzate dettagliate descrizioni del luogo in cui ci si trova. Inoltre, il giocatore si ritrova realmente coinvolto nel gioco, unico artefice della propria sorte.

L'Adventure, in altre parole, è un tipo di gioco in cui vengono privilegiati il ragionamento, la riflessione e la calma piuttosto che la velocità del joystick o il colpo d'occhio.

Come funziona un Adventure

Generalmente, dopo il caricamento del programma, viene descritto (o raffigurato) il punto di partenza dell'avventura proposta con una descrizione di ciò che ci circonda. Successivamente viene richiesto il da fare. Esistono parole-chiave inserite in un vero e proprio vocabolario fornito con il package.

Tuttavia, a parte piccole differenze, le parole-chiave sono praticamente sempre le stesse per tutti i giochi e si riferiscono principalmente a comandi di movimento, di apertura, di chiusura, di osservazione e di inventario.

Spesso, anzi quasi sempre, nelle descrizioni sono celati giochi di parole o suggerimenti che si riveleranno indispensabili non solo per proseguire bene nel gioco, ma soprattutto per sopravvivere ai primi pericoli.

Fin qui nulla di speciale o di particolarmente difficile. Purtroppo, però, la maggior parte degli Adventures è in lingua inglese, per cui sarà il caso di prestare ancora più attenzione alle descrizioni con un buon vocabolario a portata di mano.

Non descriveremo dettagliatamente tutti gli Adventures sul mercato, ma parleremo solo di Zork che, stando al numero di unità vendute, è considerato uno dei migliori in commercio.

Al momento esistono tre versioni di Zork (Zork I, Zork II e Zork III); ciò che seguirà dovrà ritenersi valido solo per il primo della serie.

Zork I

E' un package che si trova solo su floppy perchè, durante lo svolgimento del gioco, si verificano numerosi accessi al disco per attingere informazioni relative allo sviluppo degli eventi.

Zork è un gioco la cui risoluzione richiede mediamente 35-40 ore. Questo fatto, però, non deve spaventare i nostri lettori perchè non si è tenuti a risolverlo tutto in un'unica "session", ma è ampiamente previsto dal programma il salvataggio della posizione raggiunta. Ciò significa che in qualsiasi momento è possibile abbandonare il gioco per riprenderlo successivamente al punto esatto in cui lo si era interrotto

La considerevole lunghezza del viaggio deve essere un incentivo ad accettare la sfida di Zork che, per un bel pezzo, vi farà compagnia, proprio come un buon libro o una rilassante partita a scacchi. Ci sembra inutile elencare i comandi possibili perchè sono chiaramente descritti nel manuale d'uso. Ci limiteremo, invece, a dare qualche consiglio utile che servirà per "campare" un po' più a lungo.

Guida all'uso di Zork

- Mappare l'Adventure a parte su un foglio di carta, cioè disegnare con dei quadrati le varie stanze visitate segnando le possibili uscite e gli oggetti eventualmente trovati.

- Mai, ripetiamo **mai,** restare al buio perchè così vi esporreste al pericolo di essere divorati da un (o una) Grue, una demoniaca presenza che affolla i posti bui o male illuminati e che predilige gli avventurieri per i suoi pranzi.

- Se vi avventurate nel labirinto e a un certo punto, a furia di girare a vuoto, con un comando di UP vi trovate in compagnia di uno scheletro e di una borsa piena d'oro, filatevela con la borsa e non toccate per nessuna ragione lo scheletro, pena la perdita di tutti i vostri averi.

- Quanto detto finora vale anche per un incontro con un uomo armato di stiletto che vi potrebbe distrarre, se non uccidere, mentre un non ben identificato individuo vi sottrae ciò che è in vostro possesso.

- Il primo dei 20 tesori che dovete recuperare si trova poco lontano dalla casa dove ha origine l'avventura, esattamente su un albero (con rami bassi vicini al terreno) su cui arrampicarsi col comando UP. Si tratta di un uovo incrostato di diamanti: prendetelo e portatelo in casa, per deporlo nella cassa del soggiorno.

- Prima di muovere verso luoghi che reputate pericolosi e, comunque, ogni 40-50 mosse, salvate la vostra posizione così da cautelarvi verso incidenti più o meno mortali che vi costringerebbero a ritornare al punto di partenza o addirittura a ricominciare tutto da capo spegnendo il computer e ricaricando il programma. Questo però si verifica solo nel caso moriate più di tre volte.

Servizio informazioni Centri di riparazione

Molti lettori ci chiedono spesso gli indirizzi dei centri di riparazione Commodore. Siamo lieti di pubblicare, qui di seguito, i nominativi ufficiali dei cent ri di riparazione forniti dalla stessa Commodore Italiana.

Ricordiamo, inoltre, che è stato istituito un centro di informazioni, presso la Commodore Italiana (tel.02/ 61.83.22.45) che risponde tutti i pomeriggi (da lunedì a venerdì dalle ore 14-18).

E' possibile richiedere informazioni di qualunque tipo sui prodotti originali Commodore sia hardware che software.

Commodore Service Center

Centri di assistenza e riparazione

ANCONA
CAR - Via Primo Maggio 10 - 60020 - Tel. 071/804488.

BARI
VIGA - Via D. Morea 42 - 70124 - Tel. 080/413766.

BOLOGNA
DR. SAX - Via D. Creti 77/C - 40126 - Tel. 051/352539.
MASER - Via Di Corticella 177 - 40128 - Tel. 051/326420.

BOLZANO
ELECOMP - Via Druso 52/A - 39100 - Tel. 0471/42128.

CAGLIARI
GENERAL PROGETTI - Centro Commerciale Rejna - V.le Monastir 110 - 09100 - Tel. 070/283291.

CARPI
CENTRO RIPARATORI - Via Remesina 226 - 41012 - Tel. 059/686874.

CASALNUOVO
GAMMA ELECTRONICS - Via Naz. delle Puglie km 36,266 - 80013 - Tel. 081/8421927.

CATANIA
CAT ELETTRONICA - Via Ravenna 7/A - 95100 - Tel. 095/438670.

CATANZARO
HARDWARE SYSTEM - Via Carlo V 174 - 88100 - Tel. 0961/72500.

CINISELLO BALSAMO
ASCO ITALIA - Via T. Signorini 24 - 20092 — Tel 02/6186102 - 6185546.

COSENZA
SERVICE CENTER - Via Parisio 25 - 87100 - Tel. 0984/75741.

CREMONA
COMPUTER SERVICE CENTER - Via Genala 21 - 26100 - Tel. 0372/ 435861.

FIRENZE
PAOLO PAOLIERI - Via Gordigiani 32/A - 50127 - Tel. 055/362069.

GENOVA
SIRAGUSA - Via Milano 41 - 16126 - Tel. 010/261655.

LECCE
G.M. ELETTRONICA - Via di Casanello 61 - 73100- Tel. 0832/27174.

MESTRE
SISTEL - Via Decorati al Valor Civile 67 - 30100 - Tel. 041/935332,

MILANO
CATME - Via Severoli 9 - 20147 - Tel. 02/4152962.
TECHNOLOGY TEAM - Via Imperia 23 - 0142 - Tel. 02/8467304.

NAPOLI
ELECTRICAL ENGINEER - Via Supportico Lopez 5/A - 80137 - Tel. 081/ 444444/2931108.

PADOVA
CARPANESE ELETTRONICA TELECOMUNICAZIONI - Strada VII Martiri 101 - 35100 - Tel. 049/624160.

PALERMO
CO.AS. INFORMATICA - Via R. Mondini 3 - 90143 - Tel. 091/295209.

PERUGIA
H.C.H. - Via R. D'Andreotto 49 - 06100 - Tel. 075/753353.

PESCARA
AUDIO COMPUTER - P.za Muzii 52/2 - 65100 - Tel. 085/293375

PISA
G.L.G. ELETTRONICA - Via Pietrasantina 61 - 56100 - Tel. 050/21445.

PORDENONE
AUDIO VIDEO SERVICE - Via Gemelli - 33170 - Tel.0434/961104.

POTENZA
CITRE di Lopomo - Via dell'Edilizia - 85100 - Tel. 0971/23823.

RAVENNA
S.I.R.A. Srl - Via Aniene 42 - 48100 - Tel. 0544/64223.

REGGIO CALABRIA
HARDWARE SYSTEM - Via Fiume 2 - 98100 - Tel. 0965/91696.

ROMA
COMPUTER SERVICE ITALIA - Via Baldassarre Orero 50 - 00159 - Tel. 06/ 4382252.
TECNICOMP - Via dei Georgofili 65 - 00159 - Tel. 06/5133739.

UDINE
RT ITALIA - Via Tavagnacco 89 - 33100 Tel. 0432/481339.

il club dei club

In questa rubrica troverete gli indirizzi dei Club di utenti Commodore che hanno segnalato i propri dati alla rivista. Le associazioni che intendono entrare in questo elenco possono contattare la Segreteria di Redazione di Commodore Computer Club.

Commodoriste di tutta Italia iscrivetevi! E' nato finalmente un club riservato solo alle donne! Lady's Commodore Club. Scrivete a Valentina Cristiana Baio, Casella postale n. 29 - 36016 Thiene (Vi). Ragazze vi aspettiamo numerose!

Soft Pirate Club, via P. Amedeo, 41, 74100 Taranto. Telefonare dopo le 14 al 099/22732 chiedendo di Luigi Talamo o Luca Del Piano. Quota di iscrizione L. 10.000 prima rata e poi 5.000 mensili.

Club Commodore Cielo, via San Tiziano, 7, 31020 S. Vendemiamo (Treviso). Telefonare dal lunedì al sabato dalle 17 alle 19 al 0438/777474 chiedendo di Miguel o Josè Tomasella.

Play Arcade Club, via D'Annunzio 3, 33170 Pordenone. Telefonare dalle 14,30 alle 15,30 al 0434/41092-623365 chiedendo di Matteo Forniz o Francesco Bortolin. Quota di iscrizione L. 1.500 ogni due mesi.

Grumo ComputerClub, corso Umberto I 17,70025 Grumo Appula (Bari). Telefonare dalle 19 alle 21 al 080/622083 chiedendo di Domenico Scagliola o Vito Sardone.

Club 64, via Arosio 4, 20148 Milano. Telefonare il sabato e la domenica dalle 14 alle 18 al 02/4031643 chiedendo di Massimo Minutilli o Igor Manakoff. Quota di iscrizione L. 30.000 annuali.

Commodore System Club, via Trento 14, 43036 Fidenza (Parma). Telefonare il lunedì dalle 14,30 alle 15,30 chiedendo di Matteo Aliani al 0524/84514. Quota di iscrizione L. 2.000 mensili.

Club Commodore 64, Centro Culturale San Giacomo, via San Giacomo, 35050 Laives (Bolzano). Telefonare il giovedì dalle 20,30 alle 23 al 0471/941210 chiedendo di Giaccari. Quota di iscrizione L. 5.000 annuali.

Capo d'Orlando Computer Club, via C. Colombo 73, 98071 Capo d'Orlando (Messina). Scrivere a Giuseppe Ricciardi.

Club Charly Software, via Resistenza 26, 41033 Concordia S/S (Modena). Telefonare il venerdì e il sabato dalle ore 14 alle 20 al 0535/54325 chiedendo di Carlo Vincenzi o Donatella Giubertoni. Iscrizione gratuita.

Commodore Club Borgosesia, via Cairoli 42, 13011 Borgosesia (Vercelli). Telefonare il lunedì e il sabato dalle 9,30 alle 19,30 al 0163/26666 chiedendo di Nino o Andrea Benetti. Quota di iscrizione L. 3.000 mensili (L. 30.000 annuali).

Club Diablo, via A. D'Orsi 13, 80131 Napoli. Telefonare dopo le 9,30 chiedere di Silvio Torre o Loris Capasso. Quota di iscrizione L. 2.000 mensili (20.000 annuali).

Club I Commodorissimi, via Fratelli Bandiera, 20087 Robecco S/N (Milano). Telefonare il lunedì e il martedì al 02/9470676 chiedendo di Roberto Del Nevo o Elisa Ronzio. Quota di iscrizione L. 3.000 annuale.

Club Antares Computers Department c/o Gruppo Astrofili Antares, corso Sforza, 24 48010 Cotignola (Ravenna). Telefonare la domenica al 0545/40204 chiedendo di Luigi Foschini o Roberto Baldini. Quota di iscrizione L. 15.000 annuali.

Club Amici del Commodore, via Leonardo da Vinci 85, 80010 Villaricca (Napoli). Telefonare il martedì e il giovedì al 081/8945600 chiedendo di Fusconi Antonio o Riemm Nando. Quota di iscrizione L. 5.000 mensili.

Commodore Computer Club Mazara, via Calatafimi, 5 - 91026 Mazara Del Vallo (Trapani). Telefonare dalle 15 alle 20 al 0923/949191 chiedendo di Antonino Asaro o Sandro Ferro. Quota di iscrizione L. 10.000 mensili.

Club Spec 48k Bianco, via C. Colombo, 91 89032 Bianco (RC). Telefonare dopo le 16 al 0964/911001 chiedendo di Enzo Luca o Domenico Gimondo. Quota di iscrizione L. 10.000 annuali.

Club Sixteen, corso Garibaldi 72/3 20121 Milano. Telefonare il lunedì dopo le 14 al 02/652870 chiedendo di Umberto Bittau o Stefano Piana. Quota di iscrizione L. 20.000 mensili.

Commodore User Club c/o Ivan Gentile via Domenico Fontana 27 isolato 13, 80128 Napoli. Telefonare dalle 20 alle 22 al 081/7702166 chiedendo di Antonello d'Ottavio o Ivan Gentile. Quota di iscrizione L. 7.000 bimestrali (rinnovo L. 5.000).

Sanremo Computer Club, via Matteotti 178, 18038 Sanremo (Imperia). Telefonare dalle 21 alle 24 al 0184/75687 chiedendo di Roberto Siccardi o Sergio Greggio.

Club Commodore 64, via G. Corradi 13, 00151 Roma. Telefonare dalle 9 alle 12 al 06/5370289 chiedendo di Marco Leone o Luca Intoppa. Quota di iscrizione L. 15.000 annuali.

Tank Club, via Don Giovine 56, 15100 Alessandria. Telefonare dalle 16 alle 19 al 0131/442352 chiedendo di Andrea Valentini o G. Luca Barcellona. Quota di iscrizione L. 5.000 ogni due mesi.

Club Int. Division Users, via Montenapoleone, 9, Milano. Telefonare dalle 9 alle 12,30 al 02/701657 chiedendo di Massimiliano o di Andrea.

Club Commodore 64, via Marconi 14, 40033 Casalecchio di Reno (Bologna). Telefonare il sabato e la domenica dopo le 17 al 051/575278 chiedendo di Massimiliano Grandi e Umberto Bettini. Quota di iscrizione L. 5.000 mensili.

Club Smanettoni, via Cavour,35, 21100 Varese. Telefonare dal lunedì al sabato dalle 14,30 alle 18,40 al 0332/233162 chiedendo di Giuseppe Bruni e Jacopo Giovi. Quota di iscrizione L. 1.000 mensili.

Computers "N" programs Club, via delle Albizzie, 00172 Roma. Telefonare dopo le 16,30 al 06/288368 chiedendo di Angelo Orlandi o Massimo Avenali. Quota di iscrizione L. 5.000.

SCAMBIATEVI LE LISTE

Stefano Solla, via Monsummano Terme 7, 00118 Roma. Tel. 06/5238377-536006.

Marcello Cannata, via Trebalate-Marchesa 14, 97015 Modica (RG).

Alessandro Scardamone, via Dante Alighieri 7, 03100 Frosinone.

Fulvio Tamburriello, viale Lenin 11, 40139 Bologna. Tel. 051/543429.

Roberto Bellini, via Giulio Romano 4, 46043 Castiglione delle Stiviere. Tel. 0376/639477.

Costantino Mastracco, via Carbonaro 35, 03024 Ceprano (Frosinone). Tel. 0776/94553.

Gaetano Lepre, via Meredo 6, 20030 Seveso (Milano). Tel. 0362/501543.

Guido Scalmato, via Trilussa 37, 04011 Aprilia (LT). Tel. 06/923447.

Marco Riazzola, via Gaslini 2, 20052 Monza (Milano).

Francesco Bonati, via B. Cremagnani 1, 20059 Vimercate (Milano). Tel. 039/669796.

Carlo Pezza, via Santorre di Santarosa 61, 00149 Roma. Tel. 06/5281016.

Giorgio Cerrigone, via Roma 59, 84060 San Giorgio Alb. (CS).

Renzo Sartini, via Pisana 46, 50018 Casellina (Firenze).

Carmelo Farinella, viale Teragati 190, 96100 Siracusa. Tel. 0931/39278.

Mario Lombardi, via Palmanova 209, 20132 Milano. Tel. 02/2567039.

Renato Castelnuovo, via S. Antonio 3, 23020 Montagnana (SO). Tel. 0342/380234.

Maurizio Tarsetti, largo Baccelli 11, 61100 Pesaro. Tel. 0721/67789.

Gianni Mazzesi, via Cella 329, 48020 S. Stefano (RA). Tel. 0544/573529.

Daniele Materazzi, via Del Santo 46, 53040 Acquaviva (SI). Tel. 0578/767471.

Roberto Romano, viale Regina Margherita 19, 90138 Palermo. Tel. 091/6514969.

Franco Baldizzone, corso Inglesi 247, 180138 Sanremo (IM). Tel. 0184/881719.

Fabio Donati, via Ravenna 34, 00161 Roma. Tel. 06/429266.

Cosimo Fabroni, via D. da Settignano 23, 50135 Firenze. Tel. 055/697268.

Stefano Lori, via Carlo Pascal 5, 00167 Roma. Tel. 06/6375254.

Massimo Gallinaro, via Flavio Biondo 8, 35100 Padova. Tel. 049/689473.

Riccardo Cecchi, via Bu Meliana 12, 00195 Roma. Tel. 06/354939.

Adriano Preto Martini, via Cao di Là 19/A, 37040 San Bonifacio (VR).

Eliseo Mastrangelo, via Casilina 1641, 00133 Roma. Tel. 06/6151345.

GBM Soft c/o Carmine Cucciniello, via Napoli prima traversa 9, 81059 Vairano Scalo. Tel. 0823/988032-988140.

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... Cognome ....

Via .... N° .... CAP. .... Città ....

Telefono .... Orario ....

Registrate il mio abbonamento annuale a Commodore Computer Club.

☐ Ho versato oggi stesso il canone di L. 35.000 a mezzo c/c postale n°37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

☐ Ho inviato oggi stesso assegno bancario n ....
per l'importo di L. 35.000 intestato a Systems Editoriale
Si prega di scrivere il proprio nome e l'indirizzo completo in modo chiaro e leggibile. Inviare la fotocopia del bollettino di c/c postale.

Considerando che i numeri 1, 2 e 7 sono esauriti, vogliate inviarmi i numeri arretrati .... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri, o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L.............. Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L.................. **a mezzo c/c postale n. 37952207 intestato a:**
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

STATISTICA

Non posseggo un computer ☐
Posseggo un C64 .... si ☐ ... no ☐
Posseggo un VIC 20 .... si ☐ ... no ☐
Posseggo un Commodore Plus 14 .... si ☐ ... no ☐
Posseggo un Commodore Plus 16 .... si ☐ ... no ☐
Posseggo un registratore dedicato .... si ☐ ... no ☐
Posseggo un drive 1541 .... si ☐ ... no ☐
Posseggo una stampante .... si ☐ ... no ☐
Posseggo un monitor .... si ☐ ... no ☐

COLLABORAZIONE

A titolo di prova vi invio un articolo e la cassetta.......................disco..............................
col programma che intendo proporre per la pubblicazione di cui garantisco l'originalità.

DOMANDA/RISPOSTA ....

RICHIESTA ARGOMENTI

Mi farebbe piacere che Commodore Computer Club parlasse più spesso dei seguenti argomenti:

1/ ....

2/ ....

3/ ....

4/ ....

GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .... Voto ....

B/ .... Voto ....

C/ .... Voto ....

D/ .... Voto ....

PICCOLI ANNUNCI ....

....

....

....

....

....

CERCO/OFFRO CONSULENZA ....

....

....

....

....

....

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome ______________ Cognome ______________

Via ______________ N° ______ CAP. ______ Città ______________

Telefono ______________ Orario ______________

Suppl. a Commodore Computer Club
n. 27/1986 - Reg. Trib. Milano 370/82
Resp.: Michele Di Pisa - Distr. MePe